Modeling of Fuel Injection System for Troubleshooting

ALEXANDER GEORGII-HEMMING CYON

KTH Computer Science
and Communication

Master of Science Thesis
Stockholm, Sweden 2013

Modeling of Fuel Injection System for Troubleshooting

A L E X A N D E R G E O R G I I - H E M M I N G C Y O N

DD221X, Master's Thesis in Computer Science (30 ECTS credits)
Degree Progr. in Computer Science and Engineering 300 credits
Master Programme in Computer Science 120 credits
Royal Institute of Technology year 2013
Supervisor at CSC was Johan Karlander
Examiner was Anders Lansner

TRITA-CSC-E 2013:008
ISRN-KTH/CSC/E--13/008--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.kth.se/csc

Abstract

With the technology progressing further, making heavy duty vehicles more complex, more computerized, it becomes necessary to update the troubleshooting process of such vehicles. From the vehicles´ computers, diagnosis trouble codes can be extracted, informing the mechanic about the health of the vehicle. Using those codes, together with physical observations, as input for a diagnosing software, the software can give educated troubleshooting advice to the mechanic. Such diagnosing software requires a model of the vehicle or one of its system, which mimics the behavior of the real one. If there would be a one-to-one correspondence between observations and diagnosis, the model would be completely accurate. However, no such one-to-one correspondence exists. This makes the system non-deterministic. therefore the model has to be constructed using another approach. This master thesis presents a statistical model of a fuel injection system called XPI. The XPI-system is modeled using a statistical model called a Bayesian network which is a convenient way to model a non-deterministic system. The purpose of this model is to be able to make diagnosis of the XPI-system, but also to ease the understanding of the dependency between components and faults in the XPI-system. The model is also used to evaluate detectability and isolability of faults in the XPI-system.

Sammanfattning

Modellering av bränslesystem för felsökning

Då teknologins utveckling gör tunga fordon mer komplexa, mer datorberoende, blir det nödvändigt att modernisera felsökning av dessa fordon. Från fordonens datorer kan felkoder avläsas. Dessa felkoder meddelar mekanikern om fordonets skick. Felkoderna, i kombination med fysiska observationer, kan användas som indata till en diagnostiseringsmjukvara, som kan förse mekanikerna med kvalificerade felsökningsråd. En sådan diagnostiseringsmjukvara kräver en modell av fordonet, eller ett delsystem av det, vilken modellerar beteendet av det riktiga systemet. Om det skulle finnas en ett-till-ett mappning mellan observationer och diagnoser, skulle modellen ha fullständig precision. Dessvärre finns det ingen sådan ett-till-ett mappning. Modellen måste således konstrueras med en annan metod. Detta examensarbete presenterar en statistisk modell av ett bränslesystem kallat XPI. Denna typ av statistiska modell kallas för ett Bayesianskt nätverk, som är lämpligt att använda vid modellering av icke-deterministiska system. Syftet med denna modell är att den ska kunna diagnostisera XPI-systemet samt underlätta förståelse för beroendet mellan komponenter och fel. Modellen kan också användas för att utvärdera urskiljbarhet och isolerbarhet hos fel i XPI-systemet.

Acknowledgement

Håkan Warnquist, YSNS, for guidance and feedback.

Johan Karlander, KTH, for your input.

Anders Lanser, KTH, for help and feedback on the report.

Johan Svensson, YQN, for giving be detailed information about the XPI-system and feedback about my model.

Jonas Biteus, YSNS, for showing me around at Scania and showing me engines and the XPI-system physically.

Joe Mohs, NMKB, for giving me detailed information about XPI.

Mikael Åkerfelt, YSNB, for giving me the basic knowledge of the components and faults of the XPI-system.

Frida Larsson, TIKAB (contractor), for doing the illustrations of the engine and cylinders.

Isabella Chowra, my wonderful girlfriend for help with graphics.

Contents

I Introduction

1 Background **1**
- 1.1 Problem formulation . . . 1
  - 1.1.1 Goal . . . 1
  - 1.1.2 Thesis delimitations . . . 2
- 1.2 Thesis Outline . . . 2

2 Theory **3**
- 2.1 The diesel engine . . . 3
  - 2.1.1 The combustion process. . . . 4
  - 2.1.2 Fuel injection . . . 5
- 2.2 Basic calculus of probability . . . 6
  - 2.2.1 Stochastic variables . . . 6
  - 2.2.2 Dependence and independence . . . 6
  - 2.2.3 Joint probability distribution . . . 7
  - 2.2.4 Conditional probability . . . 8
- 2.3 Bayesian networks . . . 9
  - 2.3.1 Small Bayesian network model . . . 9
  - 2.3.2 Component variables . . . 10
  - 2.3.3 Observation variables . . . 12
  - 2.3.4 Inference in Bayesian networks . . . 13

3 Methodology **15**
- 3.1 Methods for modeling . . . 15
  - 3.1.1 Creating the model . . . 15
  - 3.1.2 Analyzing and formatting the data . . . 16
- 3.2 Methods for verifying the model . . . 16

II System Modeling **19**

4 Modeling the XPI-system **21**
- 4.1 About the XPI-system . . . 21
  - 4.1.1 Components . . . 21
  - 4.1.2 The route of the fuel in the engine . . . 21
- 4.2 Model assumptions and delimitations . . . 23
  - 4.2.1 Assumptions . . . 23
  - 4.2.2 Delimitations . . . 23
- 4.3 The model . . . 24
  - 4.3.1 Component Variables . . . 24
  - 4.3.2 Observation Variables . . . 27

4.4 Generating the model 29
4.4.1 Leaky-Noisy-Or-Nodes 29
4.4.2 Limits in GeNIe 31
4.4.3 The complete model 35

III Result **37**

5 Verifying the XPI-model **39**
5.1 Correctness of the model 39
5.1.1 Test one - expert interview 39
5.1.2 Test two - case-studies 40

6 Conclusion **43**
6.1 Discussion 43
6.2 Future work 44
6.3 Final words 44

7 Bibliography **47**

A Acronyms **51**

B Notation **53**

C Pseudocode **55**

Part I

Introduction

Chapter 1

Background

It has become difficult for mechanics to troubleshoot a faulty truck due to the increasing complexity of its systems. Trucks are nowadays composed of several complex systems containing components, sensors, computers and connections between them.

Troubleshooting refers to the act of determining which component is faulty and performing an action in order to mend this fault. Identifying the fault is not trivial, since the systems are so complex. The mechanics identifies a fault by analyzing diagnosis trouble codes (DTC) generated by the sensors in the trucks system but also by doing physical observations.

It is therefore desirable to help the mechanics with the troubleshooting process, one way of doing so is to model the truck´s systems with a statistical model that "behaves" like the real system. A certain fault in one of the truck´s systems can lead to multiple DTCs and the same DTC can be derived from different faults. This makes fault determination non-deterministic and therefore we cannot model the system with a deterministic model. Instead we represent the system´s behavior with a statistical model, so that given some observations we believe with a certain probability that a certain component is faulty.

Since it is difficult for mechanics to identify the faulty component, a quick solution would be to replace for example the whole engine instead of a small part of the engine. Isolating the fault and only doing repairs or replacements of faulty components costs less than changing whole systems in terms of material cost, but might cost more in terms of time. Because of the strong correlation between time and money, a reparation system needs to weigh material cost against time cost.

In this master thesis a model of one of the truck´s system is modeled, the e**X**treme high **P**ressure fuel **I**njection system (XPI). This model will hopefully serve two purposes; to help XPI experts to understand the system even further as well as to help the supervisor of this thesis, Håkan Warnquist, in his research of automated troubleshooting using mathematical models.

1.1 Problem formulation

1.1.1 Goal

The purpose of this master thesis is to develop a statistical model of one of Scania´s systems called XPI. The XPI-system has not been modeled as thoroughly before, hence the thesis´s main contribution is to, in a correct and accurate manner, model this system. Where great difficulty lies within defining *correct* and *accurate* as well as fulfilling those definitions. In other words, the model´s behavior should mirror that of the real system's. More specifically, if the XPI-system would indicate a certain DTC or physical

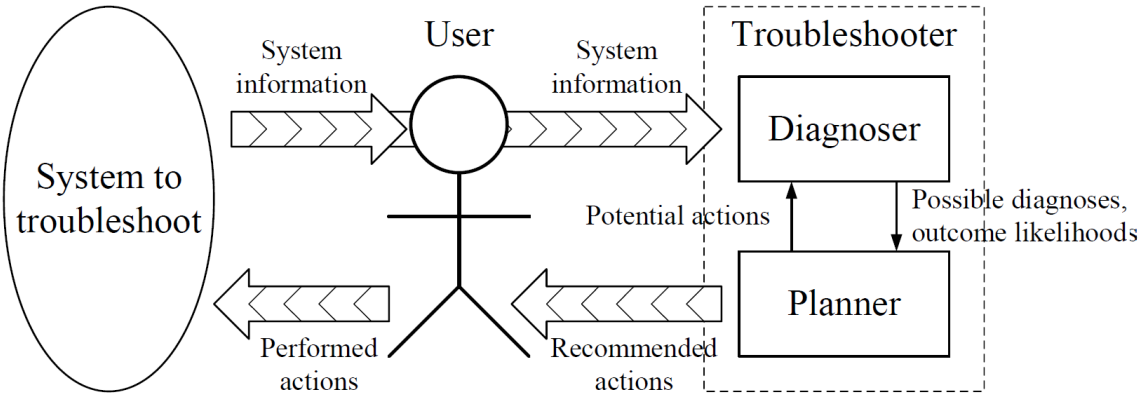


Figure 1.1: The concept of a troubleshooting framework, in this thesis only the diagnoser will be developed. The system to troubleshoot is the XPI-system, the user could for example be a mechanic, the system information consists of DTCs and physical observations. Image found in [1].

event for a certain fault, then so should the model. If a set of observations (the size of the set could be one) would lead to a fault-diagnosis by a mechanic, then the model should make the same diagnosis. Continue reading more about the behavior of the model in section *Correctness of the model* (5.1).

The model must be represented or implemented in a computer program somehow so that it is possible to interact with it. The model presented in this thesis has been represented in a graphical network program called GeNIe [2]. The model can not be created in GeNIe manually, since it is way to much work due to the complexity of the model, instead it is generated by a program, developed and presented in this thesis. This model-generating program can be reused for modeling of other systems, given that the necessary data files are available and on the correct format, described in section *Limits in GeNIe* (4.4.2).

1.1.2 Thesis delimitations

The work presented in this thesis will hopefully be useful for other systems than the XPI-systems. However, the thesis will be limited to modeling the XPI-system only. This model will have some restrictions. The XPI-system is, in fact, used in three different engine types, the 9, 13 and 16 liter engines. The 9 and the 13 liter versions does, however, differ very little in ways of design and functionality. The 16 liter version has a slightly different configuration then the other two. The XPI-system being modeled in this thesis is the 13 liter version, although the difference between it and the 9 liter version is very small[1].

1.2 Thesis Outline

First necessary theory needed to understand this thesis is explained, the reader will be introduced to how a diesel engine works in section *The diesel engine* (2.1), calculus of probability in section *Basic calculus of probability* (2.2) and what Bayesian networks are in section *Bayesian networks* (2.3). Section *Methods for modeling* (3.1) is about modeling techniques, followed by a short introduction to the model evaluation process in section *Methods for verifying the model* (3.2). After that the XPI-model is presented and thoroughly explained followed by the verification of the model. The thesis ends with a conclusion and discussion about the results in chapter *Conclusion* (6).

[1]The primary difference is that the 13 liter version has one cylinder more.

Chapter 2

Theory

The purpose of this section is to introduce the reader to the fundamental theory, necessary to understand the thesis. Two major fields are presented, firstly the reader is introduced to the basics of a diesel engine and fuel-injection. An insight of how a diesel engine works is important, because the sole purpose of this thesis is to model the fuel injection system in such an engine.

Secondly, a presentation of statistics is given, since XPI will be modeled using a statistical model, therefore it is important to have basic knowledge of the theory of statistics and calculus of probability.

A various number of concepts and their notation is presented, for the reader's convenience a summary of the notation can be found in appendix *Notation* (B).

2.1 The diesel engine

This section will explain how a diesel engine works, without the requirement of any prior knowledge of engines. A diesel engine is a combustion engine fueled by diesel[1]. When introduced, they were workhorse engines, i.e. powering heavy-duty vehicles, such as trucks, tractors, buses and trains [3]. Today nearly 70% of the automobiles on the Swedish market are powered by diesel engines [4]. The corresponding number for Europe overall is 50%, and in the global market it is predicted to be 12% globally by 2018 [5].

A combustion engine converts chemical energy in fuel to mechanical energy by burning the fuel which causes motion. The combustion, which takes place in the engines cylinders, creates a small and controlled explosion which pushes the piston in the cylinder down, giving it kinetic energy. The movement of the piston transfers force to the crankshaft making it rotate. Such a combustion takes place in every cylinder in the engine. The more cylinders, the more mechanical energy the engine creates. The volume of the cylinders also affects the power, the bigger, the more powerful and more fuel consuming. For an illustration of the cylinders see figure 2.1.

In light- to medium duty engines, such as cars and small trucks, the most common cylinder configuration is the inline-six-cylinder (also called *straight-six* engine). In heavy-duty engines, such as trucks and buses, the V6 and V8 cylinder configuration is typically used. The "V" describes the shape the cylinders constitute when having them in two planes (rows) almost perpendicular to each other, forming a V-shape, please see figure 2.1.

[1] In contrast to the "traditional" engine, fueled by **petrol**.

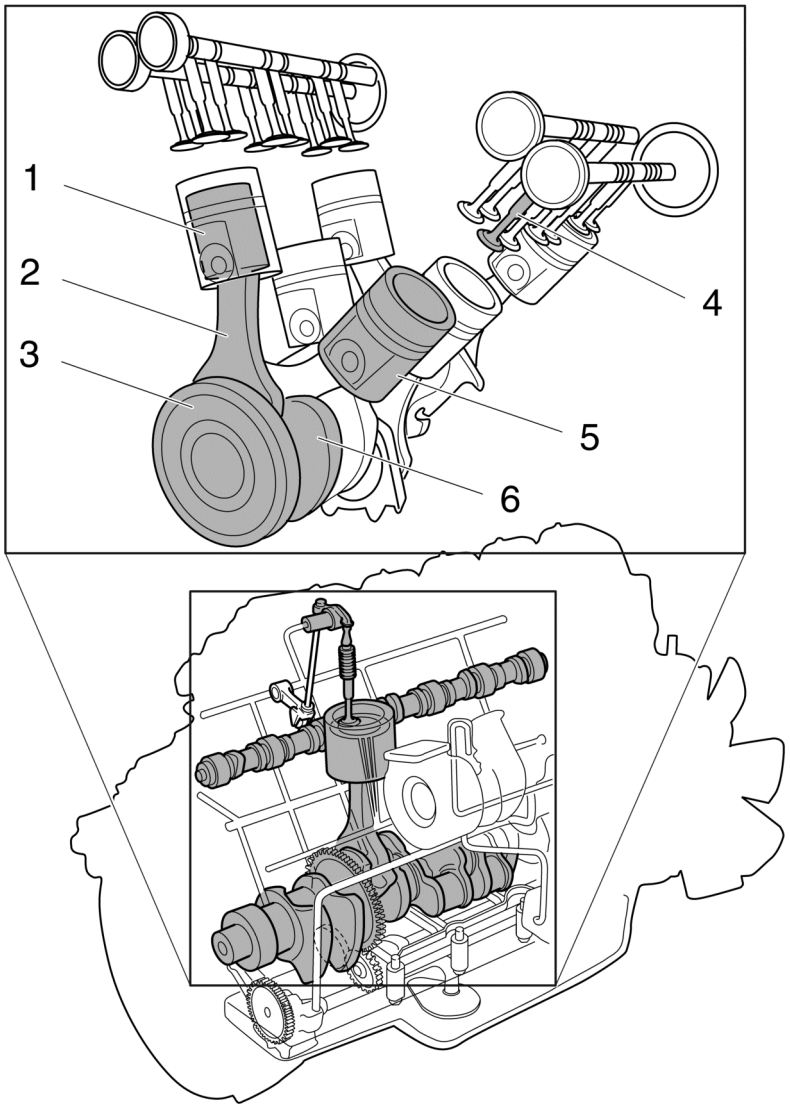


Figure 2.1: This figure shows the V-shaped configuration of the cylinders. Pistons move up and down in the cylinder, the movement of the pistons causes the crankshaft to move creating mechanical energy. 1) Piston, 2) connecting rod, 3) flywheel, 4) intake and exhaust valve, 5) cylinder, 6) counterweight

2.1.1 The combustion process.

The combustion process is classically either a two or four-stroke (step) cycle. Both the two and the four-stroke cycle is used in diesel engines. The two-stroke was more popular before, when emission control was not as strict. The two-stroke engine is cheap to produce, repair and rebuild, it is light weight but has poor exhaust emission features. Today, the four-stroke is most common [6]. Here follows a simple explanation of the combustion process of the four-stroke cycle:

1. **Intake stroke:** the piston moves down towards the bottom of the cylinder, which reduces the cylinder´s air pressure. This sucks air into the cylinder, through the air valve, please see *intake air* in figure 2.2.

2. **Compression stroke:** the piston rises, thus increasing the air pressure and temperature in the

cylinder.

3. **Power/Combustion stroke:** diesel fuel is injected with high pressure, see *fuel injector* in figure 2.2. and self-ignites due to the high pressure and temperature of the air and the fuel, this pushes the piston down with a strong force, giving the crankshaft kinetic energy.

4. **Exhaust stroke:** the exhaust valve opens and the piston returns to the top of the cylinder pushing the exhaust out and then the cycle restarts at stroke 1.

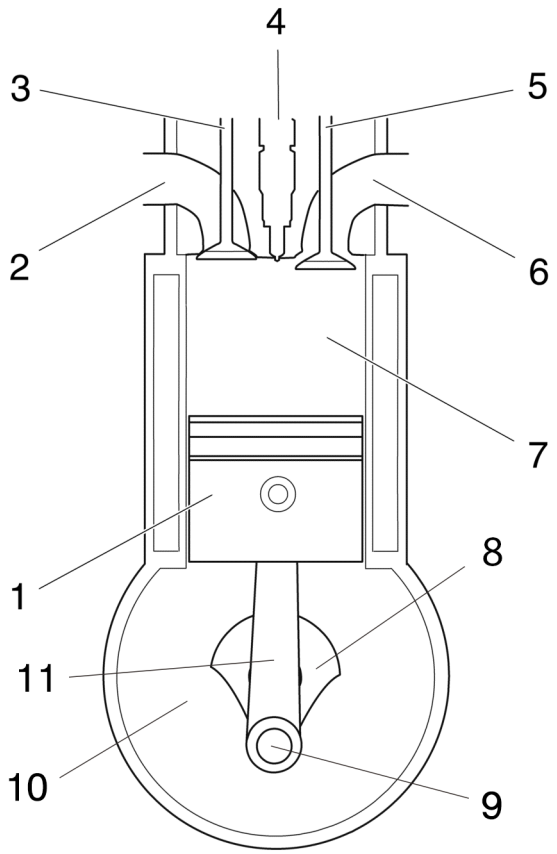


Figure 2.2: A cross section of a cylinder in a diesel engine. 1) Piston, 2) intake air, 3) intake valve, 4) fuel injector, 5) exhaust valve, 6) exhaust gases, 7) cylinder, 8) counterweight, 9) crankshaft, 10) flywheel, 11) connecting rod.

2.1.2 Fuel injection

What makes diesel engines unique is that the fuel self ignites due to the increased heat caused by the compression of the air, instead of igniting the fuel with a spark plug used in petrol engines. An important part of the diesel engine's combustion process is the fuel injection. Injecting the diesel into the cylinders with high pressure leads to a better spread of the diesel particles which increases the efficiency of the combustion [7]. Thus the fuel consumption is reduced. Another benefit is reduced exhaust gas levels and increased performance of the engine [8]. Developing a high pressure fuel injection system is therefore of interest for vehicle manufacturers. For further information about fuel injection, read the explanation of the XPI-system in section *About the XPI-system* (4.1).

2.2 Basic calculus of probability

Here follows a very brief repetition on basic probability concepts and calculation, readers with a clear understanding of calculus of probability can either skim through this section or continue reading section *Bayesian networks* (2.3). For a comprehensive introduction to statistics and probability, see [9].

2.2.1 Stochastic variables

A stochastic variable, also called *random variable*, is a variable whose value is not fixed, instead its value is described by a probability distribution. Stochastic variables are denoted with capital letters. Let the stochastic variable X_{coin} represent the outcome of a flip of a coin. If we assume the outcome to be completely random, the probability of the side facing up is equally distributed between *heads* and *tails*. The probability that stochastic variable, for example X_{coin}, has a certain value is written $P(X_{coin} = heads) = 0.5$. Stochastic variable values are often presented as probability tables, see table 2.1. Note that the probabilities must sum up to 1.0.

$P(X_{coin} = \text{heads})$	$P(X_{coin} = \text{tails})$
0.5	0.5

Table 2.1: The probability table for the stochastic variable X_{coin}.

A stochastic variable can have more than two possible outcomes. See table 2.2 for the probability distribution of a die throw, here the stochastic variable X_{die} has six different values, yet again with equal probabilities (which they do not have to be). Please note that the notation used in table 2.2, has been simplified from $P(X_{die} = 1)$ to $P(1)$. The latter is often used in the literature and will, for convenience, be used in this thesis [10, p. 3].

$P(1)$	$P(2)$	$P(3)$	$P(4)$	$P(5)$	$P(6)$
$1/6$	$1/6$	$1/6$	$1/6$	$1/6$	$1/6$

Table 2.2: The probability table for the stochastic variable X_{die}.

In both the coin flipping and the die example we have seen probability distributions where the values are discrete. However, sometimes it is needed or it is convenient to have a *continuous* probability distribution. Since the values are continuous, they can no longer be presented in a table, instead a probability function is used. The values of both X_{coin} and X_{die} can be represented with **trivial** probability functions, for example: $P(X_{coin} = i) = \frac{1}{6}, 1 \leq i \leq 6$.

When handling multiple stochastic variables it is convenient to express them as a list or a set of stochastic variables. A set of stochastic variables is denoted in capital bold font. For example let $\mathbf{X} = \{X_1, X_2, ..., X_n\}, n \in \mathbb{N}$, denote a series of die throws, where $X_i \in \mathbf{X}$ denotes the i-th die throw.

2.2.2 Dependence and independence

Stochastic variables can be related by some dependency, meaning that knowing the value of one of them affects the probability distribution of the other. An example of such dependence could be between the hour of the day and how bright it is outside. Let the stochastic variable X_{time} represent the hour of the day and $X_{sunlight}$ represent the brightness outside. X_{time} and $X_{sunlight}$ are said to be *dependent*, since it is brighter during the day and darker during the night. If two variables (or more) are not dependent, then they are said to be *independent*. Let X_{coin} be the same variable as declared in table 2.1, $X_{sunlight}$ and X_{coin} are typically independent and so is X_{time} and X_{coin}.

Definition 2.2.1 (Independence)**.** Two stochastic variables, X and Y, are *independent* if and only if, for all values x of X and all values y of Y the following equality holds: $P(X = x, Y = y) = P(X = x)P(Y = y)$. □

Definition 2.2.2 (Dependence)**.** Two stochastic variables are said to be *dependent*, if they are not independent. Thus, for all values x of X and all values y of Y the following inequality holds: $P(X = x, Y = y) \neq P(X = x)P(Y = y)$. This means that knowing the value of one of them affects the probability of the other. □

2.2.3 Joint probability distribution

Joint probability distribution describes the probability of two stochastic variables X, Y having some values simultaneously, i.e. $P(X = x, Y = y)$. This can be denoted with a shorter version: $P(x, y)$. There is also another common notation, let A denote the event that $X = x$ and B that $Y = y$, then $P(A \cap B)$ denotes that A and B occurs at the same time, however the former notation is the one used in this thesis.

Let us demonstrate with a simple example. Let $X_{even} = \{True, False\}$ be a stochastic variable denoting whether a die throw results in an even number or not, let $X_{low} = \{True, False\}$ denote whether the value is low (number 1, 2 or 3) or not. It is trivial to see that $P(X_{even} = True) = \frac{1}{2}$ and $P(X_{low} = True) = \frac{1}{2}$ respectively. X_{even} and X_{low} are *dependent* since $P(X_{even} = T)P(X_{low} = T) = 1/4$ but $P(X_{even} = T, X_{low} = T) \approx 1/6$. The joint probability distribution for the combination of those variables is shown in table 2.3.

X_{even}	X_{low}	$P(X_{even}, X_{low})$
$True$	$True$	$1/6$
$True$	$False$	$1/3$
$False$	$True$	$1/3$
$False$	$False$	$1/6$

Table 2.3: The joint probability table for two stochastic variables. The combination of $P(X_{even} = True, X_{low} = True)$ yields that the number must be even and either number 1, 2 or 3. The only number that fulfills those criteria is the number 2, i.e. only one number of six fulfills those criteria, which yields a probability of $\frac{1}{6}$. $P(X_{even} = True, X_{low} = False)$ yields an even number in the range 4-6, which is either number 4 or 6. Two possible outcomes out of six gives the probability $\frac{2}{6} = \frac{1}{3}$.

The joint probability for two independent variables are the product of their individual probability: $P(A, B) = P(A)P(B)$. Note the important difference when A and B are dependent which yields the inequality $P(A, B) \neq P(A)P(B)$.

The example presented above shows joint probability of **two** stochastic variables, however, it is possible to calculate the joint probability of a **set** of stochastic variables, $\mathbf{X}$. The calculation of the joint probability distribution of a set of stochastic variables is the product of the probability for each individual variables

in the set, or expressed mathematically:

$$P(\mathbf{X}) = P(X_0 = x_0, ..., X_{N-1} = x_{N-1}) = \begin{cases} \prod_{X \in \mathbf{X}} P(X) & \text{if } X \in \mathbf{X} \text{ is independent.} \\ \prod_{i=0}^{N-1} P(X_i | \cap_{j=0}^{i-1} X_j) & \text{if } X \in \mathbf{X} \text{ is dependent.} \end{cases} \tag{2.1}$$

2.2.4 Conditional probability

Conditional probability expresses the probability of some event A, given the knowledge that some other event B is known to have occurred, and is denoted as $P(A|B)$. The conditional probability can be calculated according to definition 2.2.3.

Definition 2.2.3 (Conditional probability)**.** The conditional probability between the stochastic variables A and B is expressed by the following equation $P(A|B) = \frac{P(A,B)}{P(B)}$, where $P(B) > 0$ and $P(A, B)$ is the probability that A and B occurs simultaneously (joint probability). If A and B are independent, then $P(A|B) = P(A)$, i.e. knowing that B has occurred does not change the probability of A. □

Since the two stochastic variables in section *Joint probability distribution* (2.2.3) are dependent, we can illustrate conditional probability using them. Knowing the value of either X_{low} or X_{even} affects the probability of the other one. Given $X_{even} = True$, the probability distribution of X_{low} changes. Since there are only one even number in the range 1-3, namely number 2, knowing $X_{even} = True$ makes it twice as probable that $X_{low} = false$ (since there are two numbers in the range 4-6 that are even, namely number 4 and 6).

Let T denote $True$, the conditional probability $P(X_{low} = T|X_{even} = T)$ is calculated as follows:

$$P(X_{low} = T|X_{even} = T) = \frac{P(X_{low} = T, X_{even} = T)}{P(X_{even} = T)} = \frac{1/6}{1/2} = \frac{2}{6} = \frac{1}{3} \tag{2.2}$$

Conditional probability can comprehensibly be visualized using venn diagrams, see figure 2.3.

A table of conditional probabilities are called conditional probability table (CPT), the CPT for the variables in table 2.3 are presented in table 2.4.

X_{low}	X_{even}	$P(X_{low}\|X_{even})$	$P(X_{even}\|X_{low})$
$True$	$True$	$1/3$	$1/3$
$True$	$False$	$2/3$	$2/3$
$False$	$True$	$2/3$	$2/3$
$False$	$False$	$1/3$	$1/3$

Table 2.4: The CPT for the two dependant stochastic variables X_{low} and X_{even}. Note that $P(X_{low}|X_{even}) = P(X_{even}|X_{low})$, since they are symmetrical.

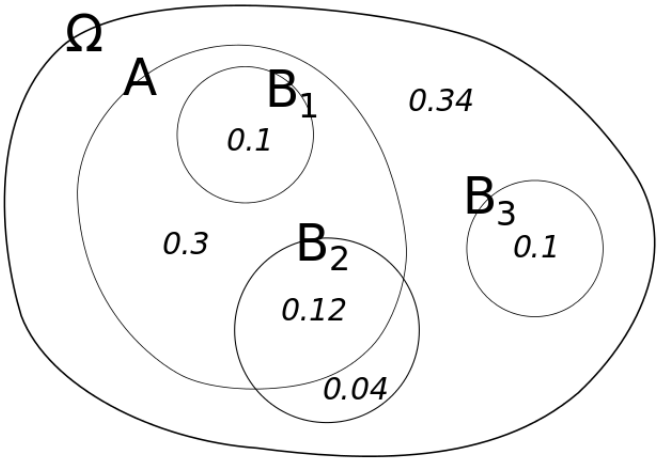


Figure 2.3: Venn diagram for illustrating conditional probability. Ω is the whole probability space ($\Leftrightarrow P(\Omega) = 1.0$), A and B_{1-3} are stochastic variables representing events. The unconditional probability $P(A) = 0.3 + 0.1 + 0.12 = 0.52$ and the conditional probabilities are $P(A|B_1) = 1$, $P(A|B_2) = \frac{P(A,B_2)}{P(B_2)} = \frac{0.12}{0.12+0.04} = 0.75$ and $P(A|B_3) = 0$. The last conditional probability is zero because it is known that B_3 has occurred, but it is not possible that A occurs at the same time, thus A and B are *independent*. As mentioned earlier, the whole probability space must sum up to 1.0, which it does: 0.52+0.04+0.10+0.34 = 1.0. Image curtsy of Wikipedia [11]. □

2.3 Bayesian networks

A Bayesian network (BN) is, a graphical model that explains dependencies and the joint probability distributions between stochastic variables. A BN can be used as a statistical model of a system, be it an environmental, technical or economical system. This thesis will use the definition of BNs given in [12].

Definition 2.3.1 (Bayesian network)**.** A Bayesian network is a triple $B = \langle \mathbf{X}, \mathbf{E}, \Theta \rangle$ where $\mathbf{X}$ is a set of stochastic variables and $\mathbf{E}$ is a set of directed edges between the stochastic variables such that $(\mathbf{X}, \mathbf{E})$ is a directed acyclic graph. The set Θ contains parameters that define the conditional probabilities $P(X|pa(X))$ where $pa(X)$ are the parents of X in the graph. □

In the graph that constitutes the BN, the stochastic variables are represented as nodes, in this thesis the nodes are drawn as circles. An edge between two nodes, is drawn as a directed arrow.

2.3.1 Small Bayesian network model

This section introduces the reader to modeling using a BN. A simple model will be presented with the goal of easing the understanding of how BNs are used to model real world objects, systems and scenarios. A BN model of a bicycle wheel describes the fault probabilities and dependencies between the wheel´s components. A bicycle wheel consists of a metal rim with spokes, a tube and a tire. The tube can be faulty, i.e. some sharp pointy object can penetrate the tire and punctuate it. Holes on the tube affects its air pressure; the air pressure **depends** on the tube. This means that the believed state, *mode*, of the air pressure depends on the information about the mode of the tube. Mounted on the tube is an air valve, which may be leaking air.

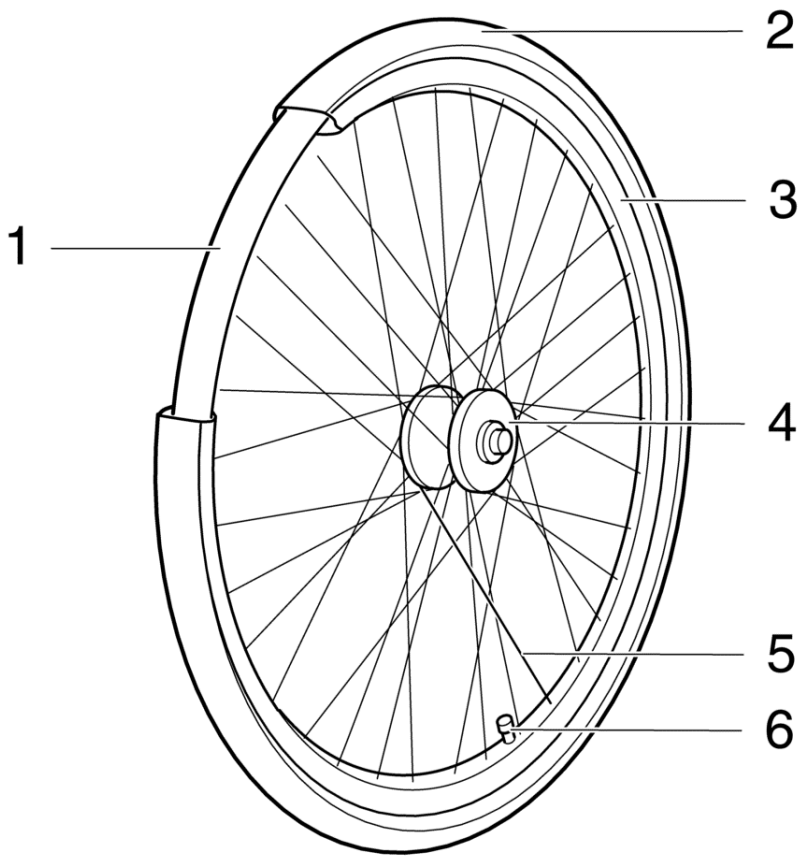


Figure 2.4: Overview of a bicycle wheels components, 1) tube, 2) tire, 3) rim, 4) hub, 5) spoke, 6) air valve.

2.3.2 Component variables

We would somehow like to represent the *mode* of the parts of the wheel in a model, or to be specific, the **belevied** mode. This is done by representing them as stochastical variables.

Definition 2.3.2 (Component variable)**.** A component variable is a stochastic variable. The values that a component variable (CV) holds are called *fault modes*. Each CV have at least two fault modes, one of them is the *non-faulty* (NF) mode[2], which means that the component that the CV represents/models is working correctly. Each CV have at least one *faulty* mode, with some specific name, e.g. *Electrical fault*, if the component that the CV models contains some electric circuit. Let $\boldsymbol{M}(C)$ denote the set of all possible fault modes for some component variable C. As shown in table 2.3, each possible fault mode must have a probability, and the sum of all modes must be 1.0. This can be expressed formally as follows: each mode $m_i \in \boldsymbol{M}(C)$ must have some probability such that $\sum_{i=0}^{M_c(C)-1} P(C = m_i) = 1.0$, where $M_c(C)$ is the size of $\boldsymbol{M}(C)$, i.e. the number of possible fault modes of C. □

In the wheel we model three of its components, X_{hole} holds information whether there is any hole on the tube or not, X_{valve} describes if the air valve is tightened or leaking and $X_{pressure}$ describes the

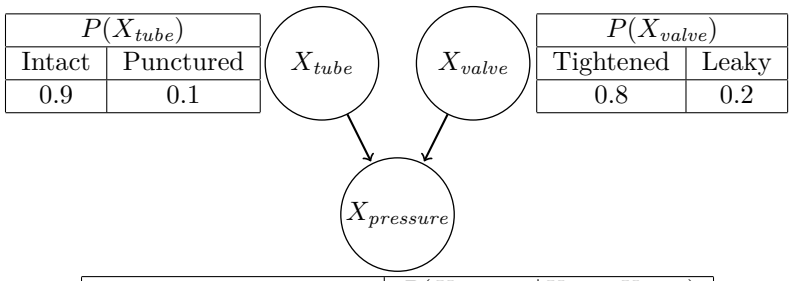
[2]One could claim that it is an oxymoron to call the mode *non-faulty* a *fault* mode.

air pressure in the tube. An overview of the variables used in the BN model of the bicycle wheel is given in table 2.5. Note that the *non-faulty* mode has been given a more specific name for the sake of comprehensibility.

Variable	Non-faulty mode	Faulty mode
X_{tube}	Intact	Punctured
X_{valve}	Tightened	Leaky
$X_{pressure}$	High	Low

Table 2.5: This table shows the three component variables in the BN model of the bicycle wheel. The *non-faulty* mode has been given a more specific name for the sake of comprehensibility.

Let $X_{pressure}$ depend on X_{hole} and X_{valve}, but the two latter be independent of each other. X_{hole} and X_{valve} have so called *a priori* probabilities, which expresses the probability for each fault modes, given no information about the system. The a priori probabilities are presented in the tables next to the nodes in figure 2.5. The table next to variable $X_{pressure}$ is its CPT.



$P(X_{tube})$	
Intact	Punctured
0.9	0.1

$P(X_{valve})$	
Tightened	Leaky
0.8	0.2

		$P(X_{pressure}\|X_{tube}, X_{valve})$	
X_{tube}	X_{valve}	High	Low
Intact	Tightened	0.95	0.05
Intact	Leaky	0.70	0.30
Punctured	Tightened	0.05	0.95
Punctured	Leaky	0.01	0.99

Figure 2.5: A small BN describing conditional probabilities of certain faults on a bicycle wheel.

The BN model of the bicycle wheel consists of the nodes (with their a priori probabilities): representing components, the arcs: representing dependency and the CPT. For a comprehensive introduction to modeling using BN see [13].

The a priori probabilities are typically calculated from data, but please note that the values in this bicycle wheel example are nothing but guessed values. More importantly, the values in the CPT of $X_{pressure}$ are also guessed and can not be calculated given the a priori probabilities of its parents. Given the tables in figure 2.5, we can calculate the probabilities of all possible scenarios, triples $\langle X_{pressure}, X_{tube}, X_{valve} \rangle$, in the wheel model. The calculated probabilities are presented in table 2.6.

From table 2.6 we can derive the total probability for high/low air pressure, by adding the four different combination which results in high and low air pressure respectively, which is done and presented in table 2.7. Even though the values in this table are nothing but calculation based on the original values in figure 2.5, it is one of the most relevant tables, from which we can conclude that the probability that the air pressure in the tube is high is 0.8142, given no information about the condition of the system.

$X_{pressure}$	X_{tube}	X_{valve}	$P(X_{pressure}, X_{tube}, X_{valve})$
High	Intact	Tightened	0.684
High	Intact	Leaky	0.126
High	Punctuated	Tightened	0.004
High	Punctuated	Leaky	0.0002
Low	Intact	Tightened	0.036
Low	Intact	Leaky	0.054
Low	Punctuated	Tightened	0.076
Low	Punctuated	Leaky	0.0198

Table 2.6: CPT over different combination of the modes of the three stochastic variables. The rightmost column shows the probability for the triple $\langle X_{pressure}, X_{tube}, X_{valve} \rangle$. The probability in the first row is calculated by: $P(Intact)P(Tightened)P(High|Intact, Tightened) = 0.9 * 0.8 * 0.95 = 0.684$. And the last row is calculated by: $P(Punctuated)P(Leaky)P(Low|Punctuated, Leaky) = 0.1 * 0.2 * 0.99 = 0.0198$. etc.

$P(X_{pressure}\|X_{tube}, X_{valve})$	
$P(High)$	$P(Low)$
0.8142	0.1858

Table 2.7: This table groups the probabilities depending on the $X_{pressure}$ variable. Where the value in the left column is the sum if the first four rows in table 2.6 are $P(High) = P(High, Intact, Tightened) + P(High, Intact, Leaky) + P(High, Punctuated, Tightened) + P(High, Punctuated, Leaky) = 0.684 + 0.126 + 0.004 + 0.0002 = 0.8142$ and $P(Low)$ is calculated analogously, with the four last rows.

2.3.3 Observation variables

To identify which of the system´s components is faulty and make a diagnosis, observations has to be made on the faulty system. It can either be a physical observation made by the mechanic or DTCs provided by the XPI-system´s computer system called *engine management system* (EMS)[3]. In the bicycle wheel example such observations can be examination of the tube and the valve, with the purpose of determining if the tube is punctuated and whether the valve is tightened or not.

Definition 2.3.3 (Observation variable)**.** An observation variable is a stochastic variable. An observation variable (OV) can only have two different modes, either *indicating* (ind) or *non-indicating* (nind). Edges are directed from CVs to OVs. An OV may have multiple edges to it, i.e. multiple component nodes as parents. □

As mentioned, $pa(O_i)$ denotes the parent list for the observation variable O_i, i.e. a list of CVs.

$\boldsymbol{M}(C_j)$ denotes the set of fault modes of the component variable C_j, then there is a conditional probability that parent to O_i, C_j, is in the mode m_k.

$$P(O_i = ind|C_j = m_k),\ \forall C_j \in pa(O_i)\, \forall m_k \in \boldsymbol{M}(C_j)$$

Thus there is a CPT for each OV, with $\prod_{j=0}^{pa_c(C_j)-1} 2 * \boldsymbol{M}(C_j), \forall C_j \in pa(O_i)$, where $pa_c(C_j)$ is the number of parents to O_i. The factor 2 comes from that each of OVs O_i´s modes (ind/nind) must have a conditional

[3]A type of *electronic control unit* (ECU)

probability[4]. The number of probabilities can be generalized to the asymptotic upper bound:

$$\mathcal{O}(2M^C) = \mathcal{O}(M^C)$$

Where $C = pa_c(O_i)$ and $M = max(\boldsymbol{M}(C_j)), \forall C_j \in pa(O_i)$ In other words the maximum fault mode probability of O_i´s parents raised to the power of the parent count for that OV. In other words, the size of the CPT grows exceptionally, which is an unwanted complexity.

The exponential growth of CPTs for OVs, explained in definition 2.3.3 is visualized in table 2.8. In section *Generating the model* (4.4) it is shown that the exponential growth poses a big problem for the XPI-model, as a solution to this, another representation of observation variables is presented, with a probability table that has polynomial growth.

C_{01}	m_0						NF					
C_{02}	m_0			NF			m_0			NF		
C_{03}	m_0	m_1	NF	m_0	m_1	NF	m_0	m_1	NF	m_0	m_1	NF
ind	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}
$nind$	$\bar{P}_1$	$\bar{P}_2$	$\bar{P}_3$	$\bar{P}_4$	$\bar{P}_5$	$\bar{P}_6$	$\bar{P}_7$	$\bar{P}_8$	$\bar{P}_9$	$\bar{P}_{10}$	$\bar{P}_{11}$	$\bar{P}_{12}$

Table 2.8: This is an example of a CPT for an observation variable with three parents, note that the fault modes m_i for a row belongs to the component variable written in the leftmost column. The probabilities P_i are numbers between zero and one, and $\bar{P}_i$ denotes the complementary probabilities, i.e. $1 - P_i$. Note that C_{01-02} have two fault modes, m_0 and NF, and C_{03} has three, thus the number of probabilities are $2 * (2 * 2 * 3) = 24 = \mathcal{O}(3^3) = \mathcal{O}(29)$. □

The model of the bicycle wheel presented in figure 2.4 earlier can be extended to include the observation nodes, and edges between them and the component nodes, representing their dependency.

In figure 2.6 three nodes have been added, those are white and represents observation variables. Please note that the component variables have been renamed from X_i to C_i, where C stands for *component*. Also note the difference between $C_{pressure}$, which models the **actual** air pressure, and $O_{pressure}$, which models the **observed** pressure. All values in the CPTs are guessed by the author, from those values the probabilities for making the observations can be calculated and are presented in table 2.9.

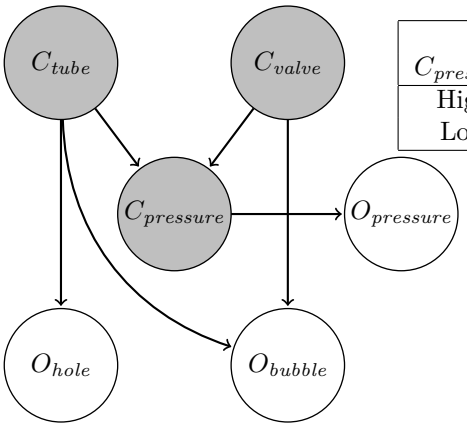
$P(O_{pressure})$		$P(O_{bubble})$		$P(O_{hole})$	
High	Low	Bubbles	No bubbles	Hole found	No hole found
0.77	0.23	0.234	0.766	0.01	0.99

Table 2.9: This table shows the probabilities for the different modes of the observation variables. The probabilities presented in this table are the probabilities to make those observations given no information about the fault modes of the component variables.

2.3.4 Inference in Bayesian networks

When a model is created, it could be used for making diagnosis. Using a model for diagnosis requires *evidence* of its variables and an *inference* algorithm. Evidence is the term for **known** statistical events, for example that C_i is in the mode m_j. Setting that evidence can be written as $C_i \leftarrow m_j$. Inference is the process of deriving logical conclusions, in this case, from statistical data [14]. Inference and evidence can be studied by looking at the example given in figure 2.6. Since some of the probabilities are conditional,

[4]This can seam redundant since the sum of $P(O_i = ind|C_j = m_k) + P(O_i = nind|C_j = m_k) = 1.0$ and $P(O_i = nind|C_j = m_k)$ therefore always can be derived from $P(O_i = ind|C_j = m_k)$. However this thesis assumes that both probabilities must be specified explicitly.



$C_{pressure}$	$P(O_{pressure}\|C_{pressure})$	
	High	Low
High	0.9	0.1
Low	0.2	0.8

C_{tube}	$P(O_{hole}\|C_{tube})$	
	Hole found	Not found
Intact	0	1
Punctured	0.10	0.90

C_{tube}	C_{valve}	$P(O_{bubble}\|C_{tube}, C_{valve})$	
		Bubbles	No bubbles
Intact	Tightened	0.10	0.90
Intact	Leaky	0.60	0.40
Punctured	Tightened	0.50	0.50
Punctured	Leaky	0.70	0.30

Figure 2.6: The bicycle wheel model updated with observation variables, and their CPTs.

and depend on the mode of O_{hole}, setting the evidence $O_{hole} \leftarrow holefound$ affects the probabilities in the model, thus the tables must be updated. The new values for the probability tables are calculated from the CPTs and the evidence, *inferred*, using some inference algorithm. Inference can be exact or approximate. For large and complex models it is often not possible to use exact inference due to the high time and/or space complexity. The model presented in this thesis is too complex for exact inference, hence an approximate method will be used. There is no room for an explanation of how inference algorithms work in this thesis. For a comprehensive introduction to approximate inference algorithms see [15].

Chapter 3

Methodology

In this chapter the methodology used in the thesis will be presented and motivated.

3.1 Methods for modeling

The model presented in this thesis uses Bayesian networks. The main reason why XPI is modeled using BN is that the XPI-system is non-deterministic and therefore it is suitable to use a BN, due to their statistical nature. Fault counts for each component are available, from which we can calculate the a priori probabilities. The set of fault modes for each component are also provided.

Furthermore, BNs are typically well suited for modeling when you have system experts available, that can describe the system and help understanding the data[16]. Using BNs have been shown to be a convenient methodology for modeling complex system, such as ecosystems[17][18][19].

Another reason why BN was used is that the provider of the project have worked with BN in his licentiate thesis[1]. Since this thesis is closely related to the supervisor's work it was natural to use BNs too.

3.1.1 Creating the model

The model will be presented in the modeling program *GeNIe*[2]. GeNIe is used to create BNs and enables to interact with the model by setting evidence, i.e. setting a node to a certain mode, after which the model is updated and presents the different diagnoses (conditional probabilities given the evidence). However it would be intractable to create this model manually, due to its size and the cardinality[1] of some of its nodes. Furthermore, there is no feature that allows automatic placement of the nodes in order to make the graph planar[2] (if possible), or rearrange the nodes´ positions such that a minimum number of edges intersects. The graph will therefore be made in a graph making program called *yED* [20], which has the wanted feature of node rearrangement. Please observe that the yED-model only contains the graph used in the BN-model, it has no probability data. This introduces the need of converting the yED-model to a GeNIe model and somehow insert the probability data.

The author of this thesis developed a Java program called *yED to GeNIe*, or Y2G for short. Y2G takes a yED graph together with XML data files as input and outputs a GeNIe model. Continue reading about Y2G in section *Generating the model* (4.4).

[1] The cardinality is the number of neighboring nodes, i.e. the number of parent or child nodes for some node.

[2] A planar graph is a graph were no edges intersect other than in the endpoint of a node.

3.1.2 Analyzing and formatting the data

The data used in the creation of the a priori probabilities comes from service workshops all over the world. Unfortunately the data[3] is not formated on any standard format, the data have been saved on different formats by different people. The reason for this is partially that the data is reported in by mechanics who have no insight in statistics and partially because the mechanics have not been instructed well enough on how they should report the data or that they simple have been ignoring any instructions. This results in noisy data. For example, a mechanic may report that component C is faulty and having the fault mode *Leakage*, when the true fault mode of C actually was *cracked or broken* (CB), which led to the **symptom** named *Leakage*, this results in an incorrect internal fault mode distribution between the fault modes of C. therefore certain fault modes has been grouped together. The fault modes for the pipes have been changed, the mode CB has been merged into *fuel leakage* (FL), since this probably is what the mechanics who reported it meant[4].

The fault frequency for the fault mode CB has been merged into *electrical fault* (EF) for the component *pressure sensor* (C_{22}), since it probably is what the mechanics meant.

Some DTCs that are very similar have been grouped together since they are caused by the same fault mode. For example, some components can lead to multiple DTCs that are caused by electrical faults, where the DTCs could be named *short circuit to ground* and *short circuit to battery*, those DTCs has been merged into one single observation variable named *electrical fault*, for each component potentially generating multiple electrical DTCs.

The data consists of number of warranty claims per component. This constitutes the fault frequency data needed to create the a priori probabilities for each component variable in the model. The probabilities are created by dividing the fault frequencies with the size of the truck population examined, denoted POP. Choosing POP as the denominator results in that the a priori probabilities will be the probabilities that a certain component is faulty given that we take any truck, $t \in POP$ randomly "from the street" and examining it. Determining the denominator is not a trivial task, since it affects the whole model, the problem of choosing the denominator is known as the *reference class problem*[21], and is a classic statistical problem. Instead of POP, the denominator could have been the set of all trucks that have been to service or only trucks that have been to service because of known faults.

Some of the components have no reported warranty claims, this would result in probability $P(C_i = Faulty) = 0, \forall i \in \boldsymbol{S}$, where $\boldsymbol{S}$ is a set of components with zero reported claims. In other words, the model would say that it is impossible for the components in $\boldsymbol{S}$ to be faulty, which is a dangerous assumption and would limit the model dramatically[5]. To solve this problem we use a common method to handle zero-probability entries called *add-one smoothing* (also called *Laplace smoothing*)[22]. It has been shown that there are more effective techniques than add-one smoothing[23], however this method meets our needs and is very simple to use. For a further reading about smoothing methods, please see[24][25].

3.2 Methods for verifying the model

The use of BNs for modeling have steadily increased during the last decades, thanks to their suitability for modeling using statistics. But few models are verified or validated. [16, p. 3064]. The difference between validation and verification might be hard to distinguish, but here is a good definition: [26]

[3]Taken from Scania´s statistics program.

[4]This has been done for components: C07-C12, C14-C19, C20, C23, C27, C32, C33, C36, C38, which all are component variables presented in table 4.2

[5]Since the model only could behave according to how the XPI-systems in the truck population that the data is taken from have been behaved.

Validation	"Are we building the right system?"
Verification	"Are we building the system right?"

This thesis will perform a verification of the model and not a validation. The verification techniques used in this thesis is *case-based evaluation.*[27, p. 284] In fact two different case studies will be performed, one of them *supervised*, i.e. an expert of the XPI-system will be present and give his expert opinion on each case result. The second is *unsupervised*, without the presence of an XPI-expert.

The verification process is explained in detail in chapter *Verifying the XPI-model* (5).

Part II

System Modeling

Chapter 4

Modeling the XPI-system

In this chapter the theory learned in chapter *Theory* (2) and chapter *Methodology* (3) will be applied on the data, creating the model of the XPI-system.

4.1 About the XPI-system

The purpose of the XPI-system is to supply the engine with the right amount of fuel at the right time and with high pressure. It increases the diesel injection pressure to around 2000 bar. The system consists of pumps, pipes, filters and nozzles, amongst other components. The purpose of this section is to give an explanation of how the system works so that the reader has a fundamental understanding of the system in prior to the modeling of it.

4.1.1 Components

Some of the components in the XPI-system are presented in figure 4.1. The description of the XPI-system in section *The route of the fuel in the engine* (4.1.2) will refer to the components presented in the list below, using the components index in parenthesis.

4.1.2 The route of the fuel in the engine

This section explains the route of the fuel in the XPI-system, giving the reader an insight on how the system works.

Fuel is sucked from the fuel tank by the feed pump (22). The fuel enters through connection (1) and gets drawn through the suction filter (21) into the engine management system (EMS) with cooler (24) via the fuel hose (2). The fuel is drawn from the EMS (24) and its cooler to the feed pump (22) via the fuel hose (3).

The feed pump increases the fuel pressure to between 9 and 12 bar and, via the fuel pipe (4), forces the fuel through the pressure filter (6). From the pressure filter the fuel flows, via the fuel pipe (5), into the fuel inlet metering valve (IMV) (19), fitted on the high pressure pump (HPP) (23).

The IMV controls the amount of fuel that flows into the HPP and is controlled by he EMS.

The HPP increases the pressure dramatically, from previous 9 to 12 bar to a maximum of 3000 bar. Via a high pressure pipe (7), the fuel continues to the accumulator (14).

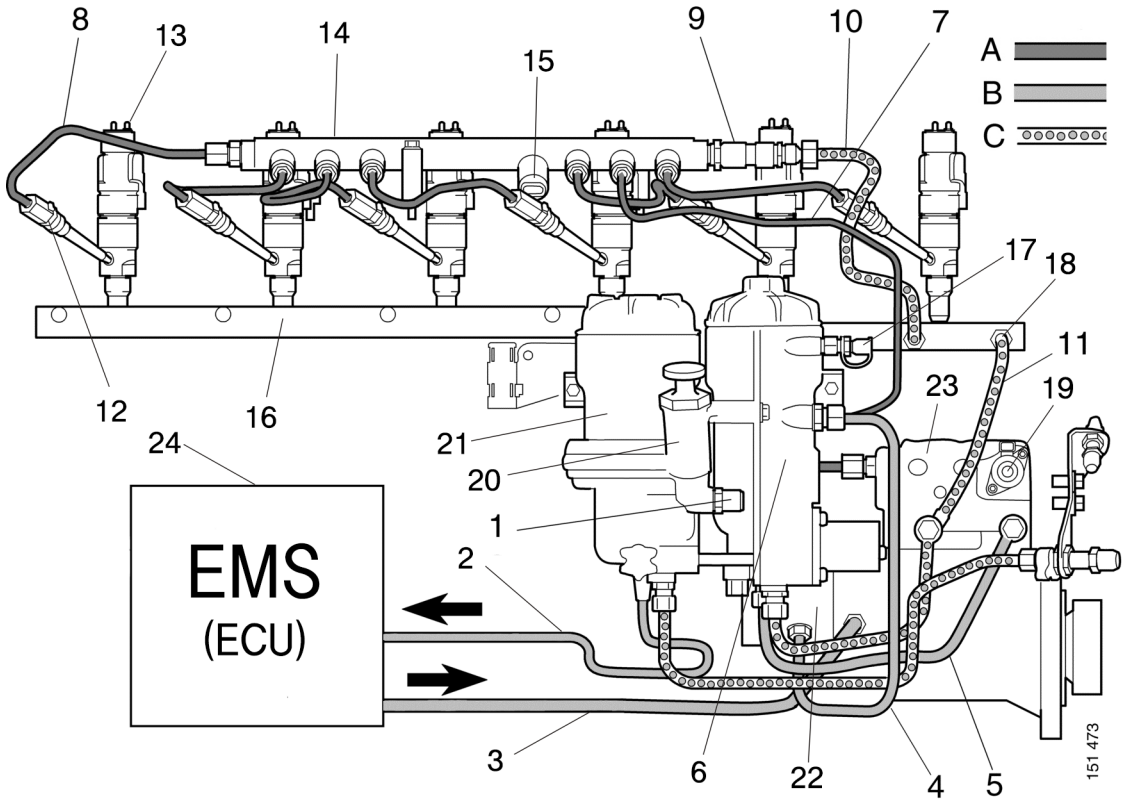


Figure 4.1: Drawing of the XPI-system, with the major components indexed. A) high pressure fuel pipe, B) low pressure pipe/hose, C) pipe for return fuel, 1) pipe from fuel tank, 2) hose to EMS cooler, 3) hose to feed pump, 4) pipe to pressure filter, 5) pipe to IMV, 6) pressure filter 7) pipe to accumulator, 8) pipe to high pressure connector, 9) safety valve, 10) pipe to fuel manifold, 11) pipe to fuel tank, 12) HPC, 13) injector, 14) accumulator, 15) pressure sensor, 16) fuel manifold, 17) bleed nibble, 18) overflow valve, 19) IMV, 20) hand pump, 21) suction filter, 22) feed pump, 23) HPP, 24) EMS with cooler

A high pressure pipe (8) connects the accumulator to each high pressure connector (HPC) (12) bringing fuel to the injectors (13). When the solenoid[1] valve in the injector is supplied with voltage, the injector opens and fuel is injected into the cylinder.

Since the XPI-system operates with high fuel pressure it is important that the fuel is free from water, because water leads to corrosion of the components. The system tolerance is tight and damaged components could cause the system to not function properly. To prevent this, water is separated from the fuel in the suction filter (21).

If a fault that results in too high pressure occurs, the safety valve (9) on the accumulator opens and the fuel is returned to the HPP via the pipe (10). The threshold for this safety procedure is > 3000 bar and drops the pressure to 1000 bar, the safety valve continues to regulate the pressure so that a pressure in the range 700-1300 bar is maintained.

Excess fuel from the injectors flows from the fuel manifold (16) back to the fuel tank via the pipe (11).

The route of the fuel is summarized in figure 4.2.

[1]A type of electromagnet.

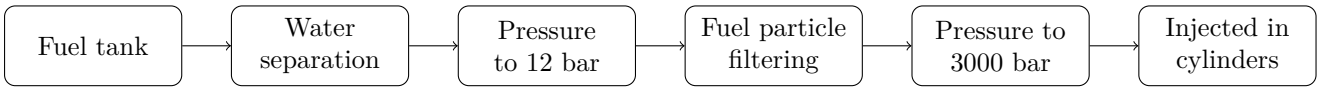


Figure 4.2: A summary of what happens in the XPI-system.

4.2 Model assumptions and delimitations

A model is, in its essence, a description of something real, therefore, disregarding how detailed the model is, there will always be some aspects and characteristics of the reality that the model will not cover. We can call this incongruence a *gap*. There will be a gap between the behavior of the real XPI-system and the model presented in chapter *Modeling the XPI-system* (4). Closing this gap requires a vastly detailed and complex model. This is not always wanted since it is difficult to get an overview of the system, making it hard to understand. Limiting the scope of the model is not only convenient, but also necessary, since it gets unmanageable to make calculations on it otherwise due to the large search space [28].

4.2.1 Assumptions

To make the modeling of the XPI-system feasible, some assumptions have to be made, otherwise the model will be too complex.

Assumption 1 (Components are independent)**.** The component variables are independent and faults occur independently. This is a strong assumption.

Assumption 2 (All components can be faulty)**.** Even though some components have a fault count of zero in the warranty claims, they have a non-zero probability of being *faulty* in the model, this is an effect of the additive smoothing mentioned in section *Analyzing and formatting the data* (3.1.2).

4.2.2 Delimitations

The XPI-system is a dynamic system that has been developed during several years. More importantly, it is still in development. The version of the system being modeled in this thesis is called *generation S7* and is installed in the heavy duty vehicles being used on the market today. The model presented in this chapter will thus only model the components, faults and observations taken from the S7 XPI-system, therefore ignoring all the updates, improvements that has been made on the XPI-system since.

The CPTs used in the XPI-model are based on statistical data derived from technical reports on vehicles during the one year warranty period sent from workshops from the whole global market. Customers are experiencing different problems e.g. due to their local whether conditions, such as temperature and humidity. another important factor is the quality of the diesel. In Sweden the diesel is very clean, almost free from water and with low paraffin[2] concentrations. However, if a vehicle is fueled with diesel with high paraffin concentrations in a country in southern Europe and enters Sweden during the winter, the paraffin often deposits into paraffin wax, clogging the fuel filters. Markets all over the world are different and in order to get a feasible and applicable model, global workshop data will be used.

Most of the pipes in the XPI-system will be included in the model. These components represents the pipes themselves but also the connection in both ends. Hoses are treated as pipes.

[2]Paraffin is a hydrocarbon compound found in diesel that can take a solid wax form if the temperature is below 37 degrees Celsius [29].

The XPI-system is not independent of other systems, the healthiness of its components may be affected by other systems in the vehicle, but it is also affected by environmental factors. The fuel filters can easily get clogged due to paraffin. As mentioned above, the fuel clogs if the fuel has a bad quality, i.e. high paraffin level in the diesel, in combination with low temperatures. These factors, as well as the other systems or parts of systems in the vehicle will not be included in this model. The reason for this is to try to limit the fault detection of the system to its components.

Since the model is suppose to reflect the reality, it can be modeled with the same limits and crassness as the reality has. In the troubleshooting manual, the mechanic is often instructed to replace whole parts of the XPI-system, consisting of several components, instead of identifying the individual component in that part. E.g. if the suction filter is broken, both that component and the pressure filter is replaced, since these components are connected and installed as a single part. therefore it is not necessary to model every single part in the XPI-system, instead the model will sometimes use one single component to represent part of the XPI-system.

4.3 The model

The data that the model is based on is sensitive to Scania, thus the data presented in this thesis are fictive. All values in tables and the model is fictive. However, the result and conclusion is based on the original data.

4.3.1 Component Variables

Deciding which components to include in the model is not an easy task, and it is one of the factors that affects the model the most.

Figure 4.1 shows the most important components in the XPI-system, but not all. The model will include those components together with some other ones. In table 4.2 all component variables in the XPI-model are presented. If a component is working correctly it is said to be *non-faulty* (NF), this mode is left out in table 4.2.

The different fault modes for the components are presented in table 4.1.

Abbreviation	Fault mode
IV	*imbalance or vibration*
FL	*fuel leakage*
CB	*cracked or broken*
EF	*electrical fault*
SC	*stuck or clogged*
WP	*wrong pressure*
EMF	*emission fault*
AL	*air leakage*

Table 4.1: A table presenting the possible fault modes for the component nodes.

Please note that **all** values in table 4.2 are fictive (randomly generated) since the real values are sensitive for Scania´s business. Since the data is fictive, the fault mode list for the components may seem strange or illogical. If the reasoning in section *Discussion* (6.1) seams illogical because of inconsistency with the data, it is because the conclusion is based on the original data.

Component variable	Description	$P(F)$[1]	Fault modes
C_{01} Accumulator (acc.)	Thick high pressure pipe/fuel rail.	5.539	{EMF:100.0}
C_{02} Feed Pump (FP)	Sucks fuel from fuel tank.	0.005	{FL:67.4, CB:32.6}
C_{03} Filter Housing	Case for C_{24} and C_{21}.	5.368	{FL:60.2, IV:26.1, EF:1.1, WP:12.7}
C_{04} Fuel Heater	Heats the fuel.[2]	3.296	{IV:38.0, WP:41.7, EF:10.1, FL:10.2}
C_{05} Fuel Manifold (FM)	Manifold for return/excess fuel.	9.496	{WP:30.6, EMF:47.8, EF:21.6}
C_{06} Hand Pump	Hand pump used for fault-diagnosis.	7.027	{SC:50.5, IV:49.5}
C_{07} HPC_1	Connector between C_{26} and C_{14}.	4.315	{IV:98.0, WP:1.4, CB:0.2, AL:0.3}
C_{08} HPC_2	Connector between C_{27} and C_{15}.	3.291	{CB:33.4, AL:52.6, SC:3.5, WP:10.5}
C_{09} HPC_3	Connector between C_{28} and C_{16}.	4.452	{EF:15.8, IV:57.8, FL:18.4, AL:8.0}
C_{10} HPC_4	Connector between C_{29} and C_{17}.	3.819	{IV:66.6, EF:10.9, EMF:18.6, SC:4.0}
C_{11} HPC_5	Connector between C_{30} and C_{18}.	6.983	{AL:1.6, EF:4.3, IV:82.1, CB:12.0}
C_{12} HPC_6	Connector between C_{31} and C_{19}.	0.801	{IV:38.6, EF:6.7, WP:52.1, EMF:2.6}
C_{13} High Pressure Pump (HPP)	Increases pressure to max. 3000 bar.	9.804	{IV:100.0}
C_{14} $Injector_1$	Injects fuel into cylinder 1.	0.510	{FL:25.9, CB:74.1}
C_{15} $Injector_2$	Injects fuel into cylinder 2.	0.510	{FL:25.9, CB:74.1}
C_{16} $Injector_3$	Injects fuel into cylinder 3.	0.510	{FL:25.9, CB:74.1}
C_{17} $Injector_4$	Injects fuel into cylinder 4.	0.510	{FL:25.9, CB:74.1}
C_{18} $Injector_5$	Injects fuel into cylinder 5.	0.510	{FL:25.9, CB:74.1}
C_{19} $Injector_6$	Injects fuel into cylinder 6.	0.510	{FL:25.9, CB:74.1}
C_{20} Inlet Metering Valve (IMV)	Controls fuel amount into C_{01}.	8.633	{EMF:1.8, SC:72.8, CB:21.2, IV:4.3}

Continued on next page…

[1]Probability that component C_i is faulty, expressed in ‰

[2]If the fuel gets cold, the risk of parrafin clogging increasesd dramatically.

Continued

Component variable	Description	$P(F)$	Fault modes
C_{21} Pressure Filter (PF)	Fine meshed particle separation filter.	4.428	{EMF:47.4, WP:50.2, SC:1.2, IV:1.1}
C_{22} Pressure Sensor	Measures the pressure.	9.644	{WP:100.0}
C_{23} Safety Valve (SV)	Opens if pressure gets too high in C_{01}.	2.287	{WP:65.0, CB:27.6, EMF:7.0, SC:0.5}
C_{24} Suction Filter (SF)	Coarse meshed water separation filter.	8.888	{EMF:47.8, EF:6.6, SC:22.1, FL:23.5}
C_{25} Venturi Housing	Venturi on C_{03}, creates suction for C_{24}.	9.260	{AL:39.9, FL:23.9, WP:26.6, SC:9.6}
C_{26} $\{acc. \Rightarrow HPC\}_1$	High pressure pipe from C_{01} to C_{07}.	7.325	{CB:72.0, EMF:9.5, EF:18.4}
C_{27} $\{acc. \Rightarrow HPC\}_2$	High pressure pipe from C_{01} to C_{08}.	1.755	{WP:79.7, SC:20.3}
C_{28} $\{acc. \Rightarrow HPC\}_3$	High pressure pipe from C_{01} to C_{09}.	8.133	{EMF:6.8, AL:24.2, WP:35.4, IV:33.6}
C_{29} $\{acc. \Rightarrow HPC\}_4$	High pressure pipe from C_{01} to C_{10}.	0.927	{EF:55.3, AL:44.7}
C_{30} $\{acc. \Rightarrow HPC\}_5$	High pressure pipe from C_{01} to C_{11}.	5.739	{SC:14.3, EF:85.7}
C_{31} $\{acc. \Rightarrow HPC\}_6$	High pressure pipe from C_{01} to C_{12}.	1.857	{AL:12.0, EF:51.8, WP:36.1}
C_{32} $\{EMSC \rightarrow FP\}$	Low pressure pipe from the EMS cooler to C_{02}.	8.252	{AL:80.5, SC:19.5}
C_{33} $\{FM \dashrightarrow Fuel\,tank\}$	Return fuel Pipe from C_{05} to fuel tank.	7.572	{SC:70.1, EMF:29.9}
C_{34} $\{FP \rightarrow PF\}$	LP from C_{02} to C_{21}.	4.158	{WP:9.7, EF:19.6, EMF:4.8, CB:65.9}
C_{35} $\{Fuel\,Tank \dashrightarrow SF\}$	Return fuel Pipe from C_{24} to C_{24}.	7.896	{EMF:71.8, FL:17.8, CB:10.4}
C_{36} $\{HPP \Rightarrow acc.\}$	HP pipe from C_{13} to C_{01}.	2.659	{AL:100.0}
C_{37} $\{PF \rightarrow IMV\}$	Low pressure pipe from C_{21} to C_{20}.	1.905	{IV:12.3, CB:60.4, EMF:27.3}
C_{38} $\{SV \dashrightarrow FM\}$	Return fuel Pipe from C_{23} to C_{05}.	5.159	{FL:5.9, AL:51.3, EMF:42.7}

Table 4.2: Table with overview of all the component variables in the XPI-model. Please note that **all**

values in this table are fictive. $A \rightarrow B$ means low pressure pipe between A and B, $A \Rightarrow B$ means high pressure pipe between A and B, $A \dashrightarrow B$ means pipe between A and B for return fuel.

4.3.2 Observation Variables

The observation variables can be classified into two different categories, DTCs taken from the EMS called Diagnosis Trouble Codes (DTC) and physical observations. The physical observations may have been made by the truck driver prior to entering the workshop, or by the mechanic during service. Such observations can be for example:

- Something smell differently than usual, maybe burnt.
- Some abnormal sound.
- Vibrations made by the engine.
- Something is visible wrong, maybe a pipe is leaking fuel.

DTCs are either *indicating* or *non-indicating*. The observation variables for the XPI-model is presented in table 4.3.

OV	Type	Description	Components
O_{01}	DTC	Accumulator pressure too low.	Leakage, C_{07-12}, C_{24}
O_{02}	DTC	Accumulator pressure too high.	Leakage.
O_{03}	DTC	Accumulator pressure too low, due to leakage in the HP part.	Leakage in HP part.
O_{04}	DTC	Accumulator pressure is excessively high.	Leakage.
O_{05}	DTC	Pressure sensor faulty.	C_{22}
O_{06}	DTC	Pressure sensor, electrical fault.	C_{22}
O_{07}	DTC	Engine over speed, possible leakage from injectors.	C_{14-19}
O_{08}	DTC	Particulate filter too hot, possible leakage inj.	C_{14-19}
O_{09}	DTC	Injectors 4-6, electrical fault.	C_{17-19}
O_{10}	DTC	Injectors 1-3, electrical fault.	C_{14-16}
O_{11}	DTC	Injection error, injectors 1-3.	C_{14-16}
O_{12}	DTC	Inj. 1 leaking, cyl. giving incorrect power.	C_{14}, C_{20}, C_{32}
O_{13}	DTC	Inj. 2 leaking, cyl. giving incorrect power.	C_{15}, C_{20}, C_{32}
O_{14}	DTC	Inj. 3 leaking, cyl. giving incorrect power.	C_{16}, C_{20}, C_{32}
O_{15}	DTC	Inj. 4 leaking, cyl. giving incorrect power.	C_{17}, C_{20}, C_{32}
O_{16}	DTC	Inj. 5 leaking, cyl. giving incorrect power.	C_{18}, C_{20}, C_{32}

Continued on next page...

Continued

OV	Type	Description	Components
O_{17}	DTC	Inj. 6 leaking, cyl. giving incorrect power.	C_{19}, C_{20}, C_{32}
O_{18}	DTC	Inj. 1, electrical fault.	C_{14}
O_{19}	DTC	Inj. 2, electrical fault.	C_{15}
O_{20}	DTC	Inj. 3, electrical fault.	C_{16}
O_{21}	DTC	Inj. 4, electrical fault.	C_{17}
O_{22}	DTC	Inj. 5, electrical fault.	C_{18}
O_{23}	DTC	Inj. 6, electrical fault.	C_{19}
O_{24}	DTC	Inj. 1, over or under fueling.	C_{14}, C_{20}
O_{25}	DTC	Inj. 2, over or under fueling.	C_{15}, C_{20}
O_{26}	DTC	Inj. 3, over or under fueling.	C_{16}, C_{20}
O_{27}	DTC	Inj. 4, over or under fueling.	C_{17}, C_{20}
O_{28}	DTC	Inj. 5, over or under fueling.	C_{18}, C_{20}
O_{29}	DTC	Inj. 6, over or under fueling.	C_{19}, C_{20}
O_{30}	DTC	IMV, electrical fault.	C_{20}
O_{31}	DTC	IMV has one or several faults.	C_{20}
O_{32}	DTC	Plausible leakage in the IMV.	C_{20}, C_{37}
O_{33}	DTC	Safety valve tripped, accumulator pressure above 2800 bar.	Leakage.
O_{34}	Visual	Fuel leakage on pipe between HPC_1 and accumulator.	C_{26}
O_{35}	Visual	Fuel leakage on pipe between HPC_2 and accumulator.	C_{27}
O_{36}	Visual	Fuel leakage on pipe between HPC_3 and accumulator.	C_{28}
O_{37}	Visual	Fuel leakage on pipe between HPC_4 and accumulator.	C_{29}
O_{38}	Visual	Fuel leakage on pipe between HPC_5 and accumulator.	C_{30}
O_{39}	Visual	Fuel leakage on pipe between HPC_6 and accumulator.	C_{31}
O_{40}	Visual	Fuel leakage on pipe between EMS cooler and FP.	C_{32}
O_{41}	Visual	Fuel leakage on pipe between FM and fuel tank.	C_{33}
O_{42}	Visual	Fuel leakage on pipe between FP and PF.	C_{34}
O_{43}	Visual	Fuel leakage on pipe between fuel tank and SF.	C_{35}
O_{44}	Visual	Fuel leakage on pipe between HPP and acc.	C_{36}

Continued on next page...

Continued

OV	Type	Description	Components
O_{45}	Visual	Fuel leakage on pipe between PF and IMV.	C_{37}
O_{46}	Visual	Fuel leakage on pipe between SV and FM.	C_{38}
O_{47}	Visual	Black smoke from exhaust pipe. Could be leaking injector.	C_{14-19}

Table 4.3: Table with the observation variables, their identifier, type and a short description of each component. In the rightmost column is a list of component variables that the observation variable is dependent on. *Leakage* can be caused by any of the following components: C_{01-03}, C_{05-06}, C_{13-21}, C_{23}, C_{25-38}. *Leakage HP part* can be caused by any of the following components: C_{01}, C_{07-19}, C_{23}, C_{26-31}, C_{36}.

4.4 Generating the model

The model was generated using the program called Y2G, developed during this thesis, as described earlier (section *Creating the model* (3.1.1)).

Y2G has a graphical user interface where the user can enter paths to models and data files and make enter some optional settings, please see figure 4.3.

4.4.1 Leaky-Noisy-Or-Nodes

As explained in definition 2.3.3, the CPTs for observation variables grows exponentially. Since some edges in the XPI-model have many parents, GeNIe would have to handle very big CPTs. O_{01} for example, has 36 parents. Modeling O_{01} as an ordinary observation node would lead to a CPT with $2^{36} \approx 6.8 * 10^{10}$ probabilities. Each probability is represented as a decimal number, taking up at least four[3] bytes of memory. This would take up a total of $2^{36} * 4\,byte \approx 275$ GB[4] of memory, just for the CPT of O_{01}.

Thus it is necessary to use some other type of node, which probability table does not grow exponentially, one such node is the *Leaky-Noisy-OR-Node* (LNON)[5] presented in [31].

Definition 4.4.1 (Leaky-Noisy-OR-Node)**.** A *Leaky-Noisy-OR-Node* (LNON) is a node with two modes, for example *indicating* and *non-indicating*. It can be set to *indicating* by any of its parent nodes, with the probability P_i, independently of the other parents. The probability that an LNON, O, is indicating is: $P(ind|\boldsymbol{C}) = 1 - \prod_{C_i \in \boldsymbol{C}} P(ind|C_i = NF)$, where $\boldsymbol{C}$ is the set of parent nodes to O. □

The LNON not only has a slower growth of its CPTs, but also a behavior wanted by most models, including the XPI-model [32]. The behavior the LNONs allow is the possibility that the node is in mode *indicating*, even though none of its parents are faulty. The probability of such false negative indications is called the *false positive rate* (FPR). It his highly desirable for a system (and thus its model) to have

[3]Double in Java is represented as a 64-bit (8 bytes) number and float is represented as a 32-bit (4 bytes) number.

[4]Equals 256 GiB [30].

[5]Called *Noisy-MAX* nodes in GeNIe.

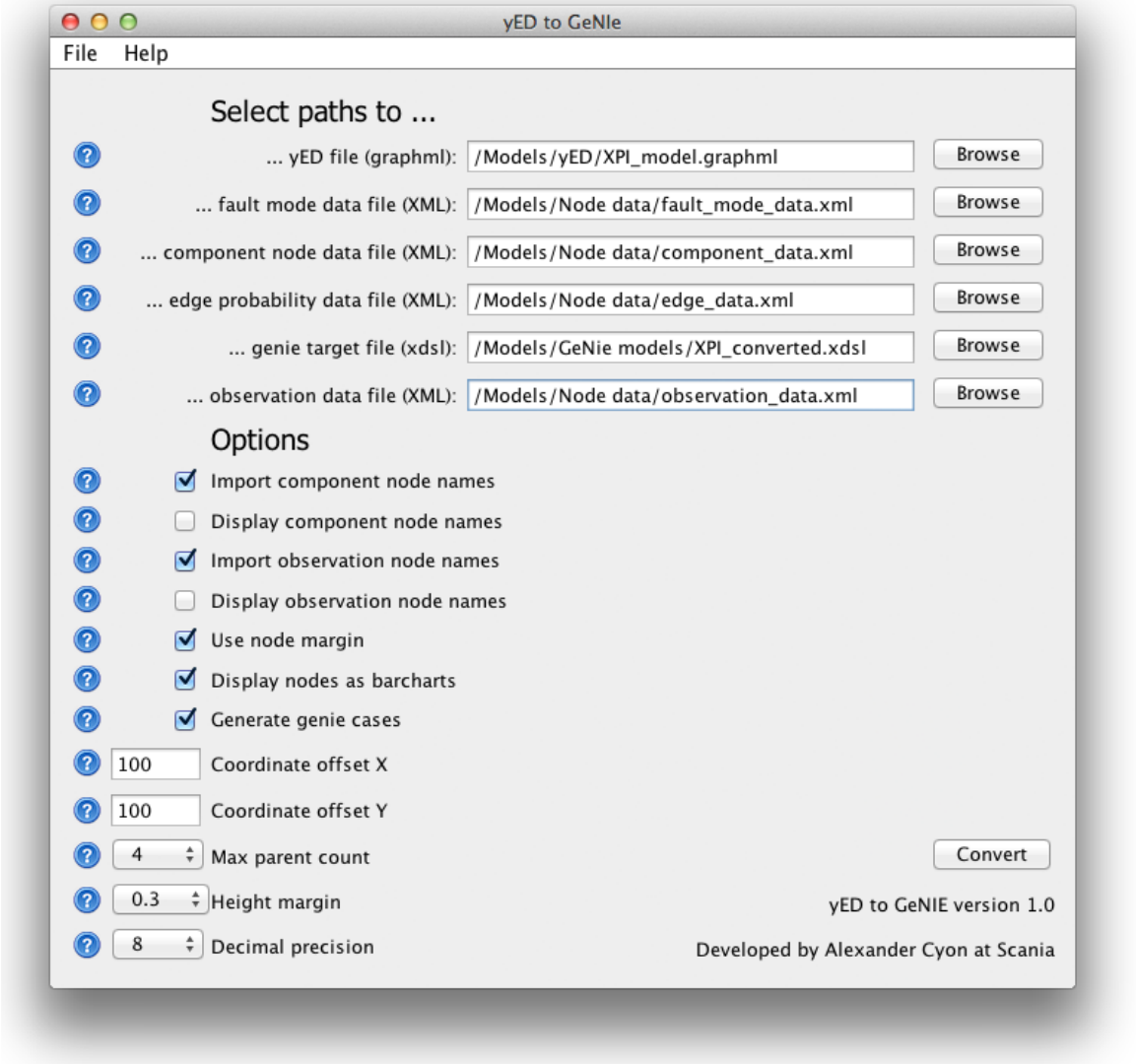


Figure 4.3: Screenshot of the Y2G programs window.

a FPR close to zero, since it makes the model even more non-deterministic and would lead to incorrect diagnosis. Even though a low FPR value is wanted, it makes the model more powerful, since it allows the model to catch casual dependencies that are outside the model, e.g. components outside the XPI-system which could affect it, as mentioned in section *Delimitations* (4.2.2).

The growth rate of a noisy node O_i is:

$$2 + \sum_{j=0}^{N-1} 2M_c(C_j) = \mathcal{O}(2 + 2\boldsymbol{M}_{max}N) = \mathcal{O}(\boldsymbol{M}_{max}N)$$

Where $N = pa_c(O_i)$ and $\boldsymbol{M}_{max}$ is the maximum fault mode count of all parent nodes. The set of fault modes for each parent node includes the mode *non-indicating*. The reason for the factor two is that each conditional probability in the CPT explicitly must define the probability of both the mode *indicating* and *non-indicating*. The reason for the constant two, is that LNON has an FPR for the mode *indicating* and

its complement, for *non-indicating*.

See table 4.4 for an example of the CPT of an LNON, note that the parents of an observation variable does not need to have the same amount of fault modes, but as stated in definition 2.3.2, all component variables have at least two modes, one of which is the NF mode.

Parent	C_{01}			C_{02}				
State	m_0	m_1	NF	m_0	m_1	m_2	NF	FPR
ind	P_0	P_1	0	P_2	P_3	P_4	0	FPR
nind	$\bar{P}_0$	$\bar{P}_1$	1	$\bar{P}_2$	$\bar{P}_3$	$\bar{P}_4$	1	$\overline{FPR}$

Table 4.4: The CPT of a noisy observation node O, where m_i is some fault mode and P_i is some probability, preferably high since it is the probability of O indicating a fault given that some if its parents, C_{01} and C_{02}, are faulty. FPR is the probability of O indicating a fault, even though none of its parents are faulty, this value is preferably very low, and in the best case scenario it is zero.

All observation nodes in the XPI-model is modeled as Leaky-Noisy-Or-Nodes, otherwise it would not be possible to create the model in GeNIe. In fact, all observation nodes are LNONs initially, but some of them are modeled a bit differently, as presented in section *Limits in GeNIe* (4.4.2).

4.4.2 Limits in GeNIe

In order to create the model in GeNIe, not only is it necessary to model observation nodes as LNONs, it is also required that no LNON has more than 20 parents[6]. Even when an LNON has more than around $13-15$ parents, GeNIe performs sluggish, inference can take several seconds. Since, observation node O_{01} has 36 parents, the graph had to be altered so that the parent count is reduced.

One of Y2G´s primary functions is that it reduces the number of parents for all nodes in the graph to *Max parent count* (MPC), which is a number chosen by the user (see figure 4.3). Y2G creates extra nodes, intermediate nodes, between the component and the observation nodes. The intermediate nodes being children to component nodes are modeled as LNONs with an FPR of zero. The observation nodes are modeled as LNON with the real, often non-zero, FPR. However, instead of conditional probabilities, $P(O = ind|C_i = m_i)$, the LNON has a probability table that creates an OR-function. The OR-function sets the observation node to *indicating* if any of its parents are indicating, otherwise *non-indicating* (see table of node O_1 in table 4.5 for an example.).

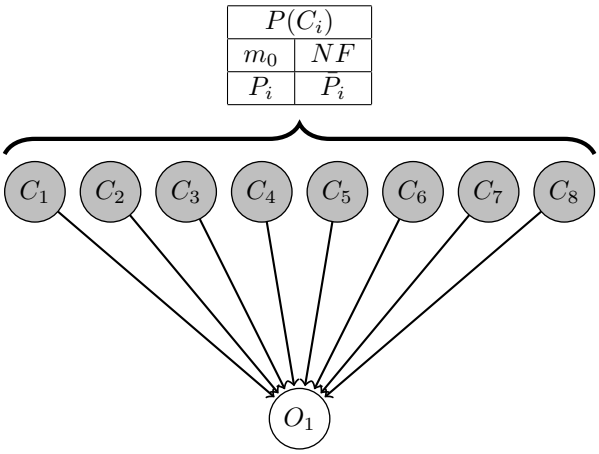
If an observation node O originally had $> MPC^2$ parents, two levels of intermediate nodes are needed. The first layer, closest to the component nodes will be LNONs with FRP or zero, the second layer consists of plain OR-nodes, without FPR, and with the bottom layer of observation nodes, modeled as LNONs with the set FPR and the OR-function characteristic. If O_i had $> MPC^3$ parents, the same thing applies, only that a third layer of intermediate OR-nodes are inserted before the layer of observation nodes. Please see figure 4.4 for an example graph prior to the Y2G conversion. The result of the conversion is shown in figure 4.5. An observation node with $> MPC$ parents forms, together with its parents, a *big tree*[7]. Those observation nodes, O_i, with an $pa_c(O_i) < MPC$ forms *small trees*. Please note that it is only the big trees in which extra nodes are inserted. The number of layers of extra nodes for the big trees is calculated by:

$$\lceil log_{MPC}(pa_c(R_i)) \rceil - 1$$

Where log_{MPC} is the maximum-parent-count-logarithm and R_i is the root of big tree T_i, an observation node that is.

[6]In fact, it is only adviced against, but the author never got the model to work with more than 20 parents, on a computer with a dual core 2.5 GHz CPU and 6GB DDR3 RAM.

[7]Inverted trees, where the root is the observation node and the component nodes are the leaves.



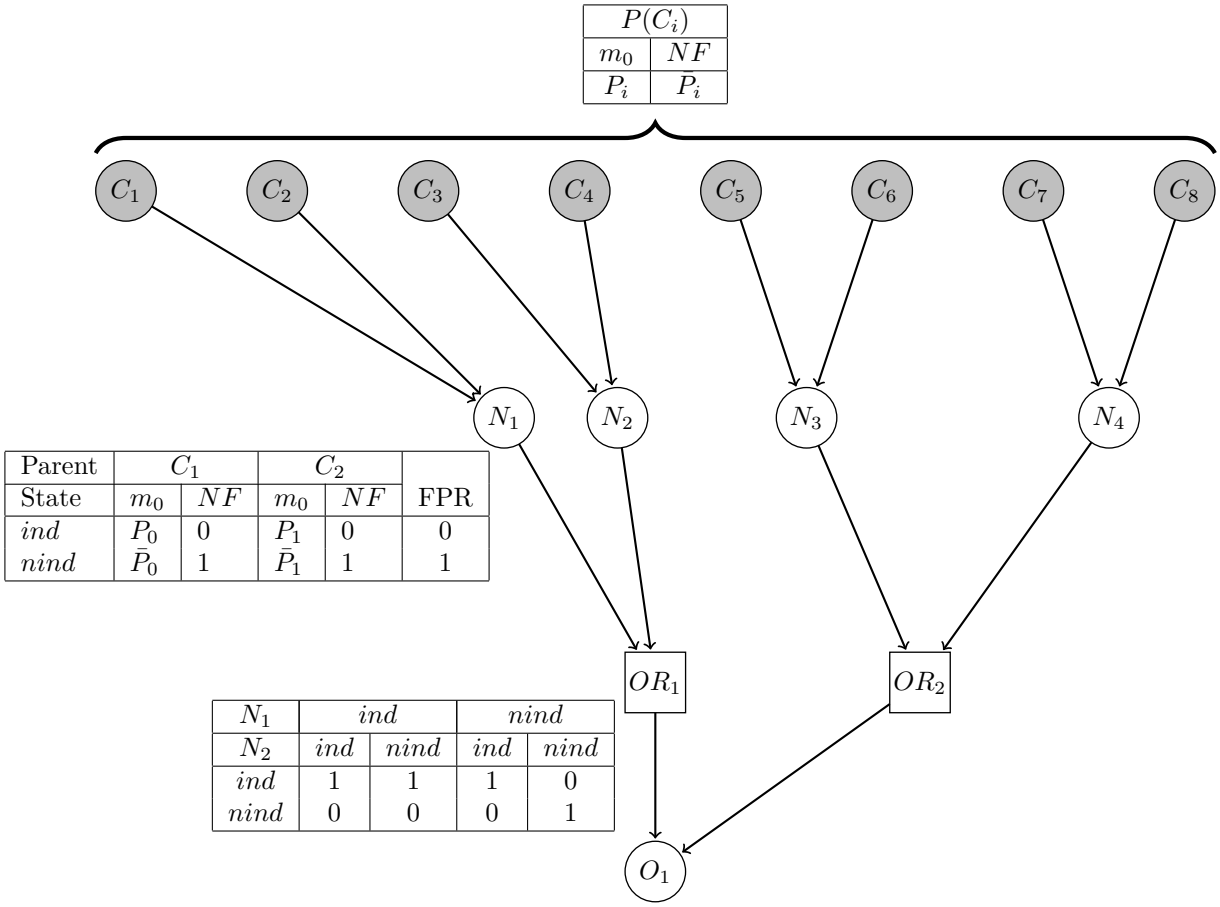
Parent	C_0		...	C_1		
State	m_0	NF	...	m_0	NF	FPR
ind	P_0	0	...	P_i	0	FPR
nind	$\bar{P}_0$	1	...	$\bar{P}_i$	1	$\overline{FPR}$

Figure 4.4: Graph G, before the conversion made by Y2G.

See appendix *Pseudocode* (C) for pseudocode of the algorithm that makes the graph conversion.

In order to create the probability tables for each node, Y2G needs data with probabilities (a priori, conditinal and FPR), but also data with list of fault modes for each component node and optionally a name for each node[8]. The data is declared in separate XML files, that is passed as input to Y2G along with the yED-graph.

[8]It is convenient to use an index as unique identifier for each node, as shown in this thesis (C_{01} and O_{01} etc.) and then declare a name for each node, that can be displayed in GeNIe.



$P(C_i)$	
m_0	NF
P_i	$\bar{P}_i$

Parent	C_1		C_2		
State	m_0	NF	m_0	NF	FPR
ind	P_0	0	P_1	0	0
nind	$\bar{P}_0$	1	$\bar{P}_1$	1	1

N_1	*ind*		*nind*	
N_2	*ind*	*nind*	*ind*	*nind*
ind	1	1	1	0
nind	0	0	0	1

Parent	OR_1		OR_2		
State	*ind*	*nind*	*ind*	*nind*	FPR
ind	1	0	1	0	FPR
nind	0	1	0	1	$\overline{FPR}$

Figure 4.5: The same graph, G, after the conversion made by Y2G, with maximum parent count set to 2. Note that the FPR in the table for N_1 (and analogously for N_2, N_3, N_4) is set to zero. Also note the OR-function in the table for node OR_1 (and analogously for OR_2) and that O_1 has a similar OR-function but has a set FPR.

4.4.3 The complete model

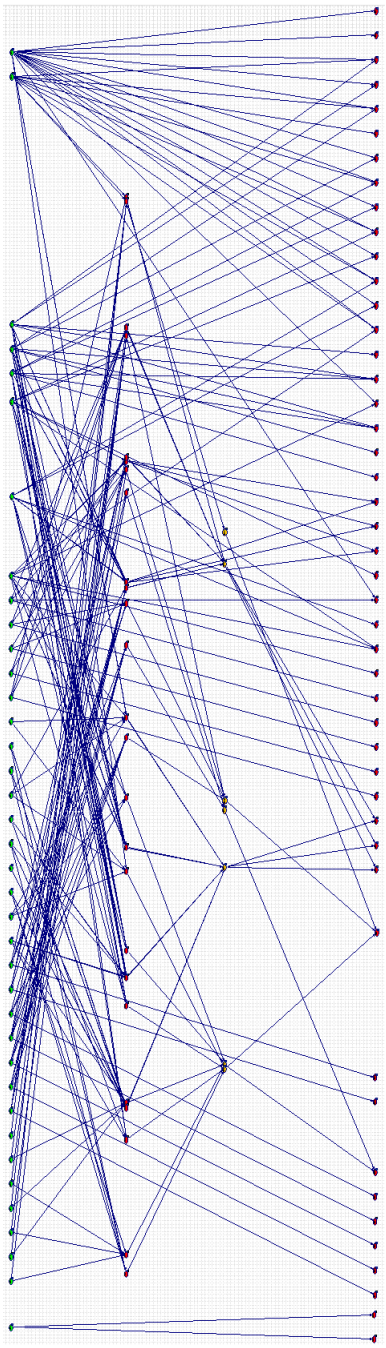


Figure 4.6: This is the complete model, with an FPR of 4. The nodes on the left side are component nodes, and on the right side observation nodes. The two layers of nodes in between the component and the observation nodes are the intermediate nodes. Many of the intermediate nodes are stacked, the are placed on top of each other. The reason for this is that Y2G places the intermediate nodes depending of the components nodes, this results in the same placement of intermediate nodes for observation nodes that have a similiar set of component nodes as parents.

Part III

Result

Chapter 5

Verifying the XPI-model

An important part of the modeling of a system is to be able to evaulate the model. There are several ways to evaluate a model, the evauluation done in this section is a *verification* that tries to determine if the model is correct.

5.1 Correctness of the model

By correctness we mean if the model of the XPI-system is realistic, if it behaves correctly. The verification of the XPI-model consists of two separate tests.

5.1.1 Test one - expert interview

The first case study was an expert interview with an engineer at Scania who has several years of experience with troubleshooting the XPI-system. For the system to be viewed as *correct*, two requirements has to be met.

Requirement one

The first step of the evaluation was studying different cases together with the XPI expert and letting him judge whether or not the model behaves correctly. By "behave correctly" we mean that the model represents the real system, by behaving reasonably much similar to the real system. Prior to the interview with the expert some formal definition of a realistic model had to be done. So the following requirement was defined:

Definition 5.1.1 (Realistic model)**.** $P(C = faulty|O)_{reality} >> 0 \quad \Rightarrow \quad P(C = faulty|O)_{model} >> 0, \ \forall C$ where $P(O|C) > 0$. □

In other words, a model is realistic if for every observation, O, examined, the probability $P(C = faulty|O = ind) >> 0$ in the model if it is in reality, for all C that can cause O.

In total 12 observation variables where examined, namely: O_{01}, O_{03}, O_{07}, O_{33}, O_{30}, O_{06}, O_{28}, O_{29}, O_{24}, O_{26}, O_{32}, O_{02}. The expert confirmed that the model was realistic according to definition definition 5.1.1. Not only did he confirm the correctness of the model, but was also impressed that the model could confirm

that the conditional probability $P(C_{20}|O_{29})$ was significantly higher than $P(C_{18}|O_{29})$, as he already had suspected[1].

Requirement two

The second part of the test was to determine that the model suggests the components in the right order.

The expert agreed that the model did so. In the model the observation O_{01} was set to *indicating*, the model then proposed the correct set of components to be faulty in an order the expert thought was likely. [2]

5.1.2 Test two - case-studies

The model was also evaluated to see that the correct observation variable was indicating for a certain fault on a certain component, and that the correct fault mode for the same component was diagnosed when setting the same observation variables to *indicating*.

Here follows two measurements of how well the model diagnoses have been defined [33].

Definition 5.1.2 (Isolable)**.** A fault mode, m_i, is **isolable** from another fault mode, m_j, if there exists some observation O such that m_i is diagnosis but m_j is not, $\forall i \neq j,\ \ m_i, m_j \in \boldsymbol{M}(O)$. □

Definition 5.1.3 (Detectable)**.** A fault mode, m_i, is **detectable** in a model if m_i is isolable from the mode NF in the model. □

The five most relevant components were studied, the components with the highest a priori probability to be faulty that is. Those were : $C_{20}, C_{23}, C_{02}, C_{13}, C_{14}$[3]. For each of those five components, it was investigated whether or not their most likely fault mode, m_{max} was isolable or not. This was done in GeNIe, by setting the evidence that C_i was in fault mode m_{max} and all other components to the mode *non-faulty*.

After all the evidence was set, it was noted which OVs had the probability $P(indicating|C_i = m_i) > threshold_{obs}$, where $threshold_{obs} = 0.99$, those OVs are denoted $\boldsymbol{O}_{ind}$. The value 0.99 was chosen to represent "certainty".

Then the evidence on C_i was cleared and all the observations in $\boldsymbol{O}_{ind}$ was set to *indicating* and all other OVs to *non-indicating*. After all the evidence was set, it was noted whether or not $P(m_{max}^{C_i}|\boldsymbol{O}_{ind}) > threshold_{comp}$, where $threshold_{comp} = 0.10$, which was chosen to represent "certainty"[4].

The model used for the results in the table 5.1 below had an MPC set to 4.

Since the model only could handle the first case study due to high complexity the case studies had to be altered. The model was too complex for an exact inference algorithm to be able to compute the conditional probabilities, probably because some nodes have a lot of parents. The alterations done was that the evidence *non-indicating* was removed for all observation nodes not in $\boldsymbol{O}_{ind}$. However, this did not help.

[1] The troubleshooting guide available for mechanics today states the reversed, which the expert suspected was incorrect, now this could be verified by the model.

[2] There is no more precise answer than that, since there is no formal way of determining the truth. The XPI-model presented in this thesis may very well be the first formal measurement of this.

[3] In fact one can say that the top 10 most relevant components was studied since C_{14} is injector 1 and injector 2-6 are symmetrical with this component.

[4] Sometimes several components are diagnosed, and several of those components fault modes, making the probabilities pretty low for each fault mode, why 0.10 is a rather high value, representing some kind of "certainty"

CV	m_{max}	$\boldsymbol{O}_{ind}$	$P(m_i\|\boldsymbol{O}_{ind})$
C_{20}	Electrical fault	O_{01-02}, O_{04}, O_{24-32}	EF: 100%
C_{23}	Fuel leakage	O_{01-04}, O_{33}	GeNIe error.
C_{02}	Cracked or broken	O_{01-02}, O_{04}, O_{33}	FL: 10%, CB: 44%.
C_{13}	Fuel leakage	O_{01-04}, O_{33}	GeNIe error.
C_{14}	Fuel leakage	O_{01-04}, O_{07-08}, O_{12}, O_{24}, O_{33}, O_{47}	GeNIe error.

Table 5.1: Table with the result from the case studies. *GeNIe error* means that no result could be obtained because GeNIe could not update the belief network, due to "too high complexity". The value of MPC was set to all values in the range 2-10, as an attempt to solve this, however, it did not matter. The second column shows the fault mode with the highest probability. The third column shows the set of observation variables having $P(indicating|C_i = m_i) > threshold_{obs}$, when the evidence $C_i = m_{max}$ was set. The rightmost column shows the diagnoses made by the model when OVs in $\boldsymbol{O}_{ind}$ was set to *indicating*.

Furthermore, another computer with 33% more memory was tested (still no evidence on *non-indicating* and with $MPC = 2$), using that computer, values could be obtained for every test case except the last one. The solution to this was to use an approximate inference algorithm. There were a variety of algorithms to chose between:

- Relevance-based decomposition.
- Polytree.
- EPIS sampling.
- AIS sampling.
- Logic sampling.
- Backward sampling.
- Likelihood sampling.
- Self-importance.

The algorithm chosen was a sampling algorithm called *Evidence Pre-propagation Importance Sampling Algorithm* (EPIS-BN). EPIS-BN was used since it is considered to be the current state of the art for sampling in Bayesian networks [34].

For an overview of inference algorithms please see [35] and [36]. Read more about stochastic sampling algorithms here [37].

Using EPIS-BN inference instead of the exact inference algorithm, we could obtain results for every test case when MPC was set to 2 on the 8gb RAM computer, with all the correct evidence set (*non-indicating* mode on all observation nodes not in $\boldsymbol{O}_{ind}$).[5]

CV	m_{max}	$\boldsymbol{O}_{ind}$	$P(m_i\|\boldsymbol{O}_{ind})$
C_{20}	Electrical fault	O_{01-02}, O_{04}, O_{24-32}	EF: 100%
C_{23}	Fuel leakage	O_{01-04}, O_{33}	FL 56.21%
C_{02}	Cracked or broken	O_{01-02}, O_{04}, O_{33}	FL: 9.12%, CB: 45.89%.
C_{13}	Fuel leakage	O_{01-04}, O_{33}	FL: 26.1%, CB: 7.72%
C_{14}	Fuel leakage	O_{01-04}, O_{07-08}, O_{12}, O_{24}, O_{33}, O_{47}	FL: 66.81%, SC: 33.19%.

Table 5.2: Table with the result from the case studies, using EPIS-BN as inference algorithm, with 100 iterations (maximum in GeNIe). But the values differed much between runs so each case was run 100 times each, for a total of $100 * 100 = 10000$ iterations. The average value for those 10000 iterations is displayed in the rightmost column.

From the table 5.2 we can deduce two important things:

1. Using EPIS-BN (10000 iterations) as inference algorithm generated similar result as the exact inference algorithm, 9.12 vs 10 and 45.89 vs 44.
2. For each case, the correct fault mode was diagnosed as the most likely fault mode, i.e. if both fault mode m_i, m_j are diagnosed, then $P(m_i|\boldsymbol{O}_{ind}) >> P(m_j|\boldsymbol{O}_{ind})$ if m_i is the correct diagnosis, where ">>" was defined as $P(m_i|\boldsymbol{O}_{ind}) = 2P(m_j|\boldsymbol{O}_{ind})$

[5]When an MPC of 4 was tested, no result could be obtained for the last test case. The reason for this is probably that four parents were too many for the inference algorithm.

Chapter 6

Conclusion

6.1 Discussion

The model presented in this thesis correctly mimics the behavior of the XPI-system. It is also compatible with the project provider´s research. The program, *yED to GeNIe*, developed by the author, and the method used for modeling can be reused to generate models of other systems, this will hopefully be helpful for the provider of the project.

The data on which the a priori CPTs are created only measures the number of times component C has been replaced, nothing else. It is important to understand that this does not necessarily mean that C has been faulty that amount of times. The mechanic that has chosen to replace C may have done so due to the simple fact that they have been instructed to, maybe because it is cheap and simple to replace C and that it *may* fix the problem. It has become clear, during interviews with experts of the XPI-system, that the pressure sensor (C_{22}) is one example of such a component. When a DTC that indicates abnormal fuel pressure is observed, the standard procedure have during a long time been to replace the pressure sensor first, even though it seldom is the culprit. Thus the model presented in this thesis is based on data derived from the incorrect fault-diagnosis done in the workshops during 2010 and 2011.

The assumptions and delimitations regarding the model may also have affected the model. Assumption 1 is a particularly strong assumption that may be true for some components and false for others. Since the fuel passes through the system in linear manner, first C_i, then C_j etc, it is quite possible that if C_i becomes faulty, the risk of C_j becomes faulty increases. For example, if the fuel heater (C_{04}) is faulty, the risk that the suction filter (C_{24}) or the pressure filter (C_{21}) clogs is increased.

In section *Analyzing and formatting the data* (3.1.2), the data was analyzed and manipulated, foremost data was grouped and split. This affected the model a lot, especially the grouping and splitting. It is very difficult to understand **how** it affected the model. The injectors was modeled individually although each injector have the same part number, and the fault count data for the injector part was divided by six. This may have been a bad modeling decision.

One of the biggest challenges when developing this model was the format in which the data was accessible. It was desired to start the modeling process by looking at the real system, and how it functions (as described in section *About the XPI-system* (4.1)). Knowing how the XPI-system functions, the next step was to identify the key component and include them as component variables in the model. But the description of the system was an abstraction of how the real system really functions. There was already a gap between the system and its description, this gap may have widened the gap between the system and the model. This poses a problem for the creation of the a priori probability tables. The reason for this is that some components actually are composed of a set of smaller parts. Thus there were no fault count

for those components, but for the parts instead. Some of those parts were used in several components, making it impossible to use the fault counts for those parts. If part p can be found in component C_1 and C_2, should the number of reported faults for p be added (merged) to C_1 or C_2 or equally divided between both? The fault counts for those parts were not merged into the fault count for those components, making the imprecise fault counts even more imprecise.

Some of the weaknesses and in the XPI-system may already have been fixed, since the version of XPI modeled is outdated by a couple of years.

Hoses were treated as pipes, which might have been a bad modeling decision, since they may "behave" differently than pipes.

The decisions made for all challenging problems above mentioned has affected the behavior of the model, and probably not always in the best of ways. It is difficult, maybe impossible to judge which decision is the right one though.

6.2 Future work

In this section suggestions for how to continue the work is presented. Those suggestions includes both improvements on the work presented but also new features and studies.

Even though the verification of the model presented in chapter *Verifying the XPI-model* (5) showed that the model behaves correctly, it is likely that it can be improved. The biggest reason to believe this is that all the conditional probabilities and the false positive rates were set by the author together with the supervisor, i.e. no expert of the XPI-systems were consulted during this process, due to limited resources. It is likely that the model will be even more accurate if those probabilities were set by an XPI expert.

The components included in the model might be changed, maybe it is better to include more, removing some, or grouping some component variables together. The injectors, component variables C_{14-19}, could for example be modeled as a single component, since their fault frequency was grouped together and then split in six equal probabilities to be modeled as six different components. The model could have modeled the dependency between fuel quality, temperature in the country and the fuel filters (C_{21} and C_{24}).

Other smoothing techniques than add-one smoothing could be used to solve zero-probability diagnosis, as mentioned in section *Analyzing and formatting the data* (3.1.2). In fact, this has not been investigated at all.

The model-generating program Y2G, programmed during this thesis maybe extended to have more dynamic input. It could e.g. be convenient to be able to enter the FPRs for the observations in the program instead of having to declaring them in XML files or enter names for the nodes.

6.3 Final words

Modeling a system as complex as XPI is difficult. The model never gets better than the that it is based on, and the data used for the XPI-model presented in this thesis is very noisy. It was, however, showed during the verification process that the model seamed to behave correct. The model was also able to confirm a theory that the expert of the XPI-system have had for some time. It is easy to understand how Bayesian networks are used for modeling, but it is hard to judge how data manipulation and modeling decisions affects the behavior of the model. There is plenty of room for improvements of the model, primary choosing the conditional probabilities for the observation variables in a better way, preferably

together with an XPI expert. The work presented in this master thesis will hopefully be of use for further research about computer assisted troubleshooting.

Chapter 7

Bibliography

[1] H. Warnquist, *Computer-Assisted Troubleshooting for Efficient Off-board Diagnosis*. PhD thesis, Linköping Universitet, 2011. Accessed online 2012-06-20.

[2] T. Loboda and M. Voortman, "Genie and smile." `http://genie.sis.pitt.edu`, 2012. Accessed online 2012-11-01.

[3] U. D. of Energy, "Just the basics: Diesel engine," tech. rep., Office of Energy Efficiency and Renewable Energy, 2003. Accessed online 2012-06-29.

[4] O. Adlercreutz, "Nära sju av tio bilar en diesel." `http://www.teknikensvarld.se/2012/05/02/30924/nara-sju-av-tio-bilar-en-diesel`, 2012. Accessed online 2012-06-29.

[5] R. Martin, "Clean diesel vehicles to represent more than 12 percent of global light-duty vehicle sales by 2018." `http://www.pikeresearch.com/newsroom/clean-diesel-vehicles-to-represent-more-than-12-percent-of-global-light-duty-vehicle-sales-by-2018`, 2012. Accessed online 2012-06-29.

[6] Wikipedia, "Two-stroke engine." `http://en.wikipedia.org/wiki/Two-stroke_engine`, 2012. Accessed online 2012-11-22.

[7] *Future and Potential of Diesel Injection Systems*, Springer Verlag, 2002.

[8] V. Hariharana and R. K. Vijayakumar, "Effect of injection pressure on diesel engine performance with sea lemon oil," *Indian Journal of Science and Technology*, 2001. Accessed online 2012-07-01.

[9] F. Dekking, C. Kraaikamp, and H. Lopuhaä, *A Modern Introduction To Probability And Statistics*. New York, USA: Springer, 2009.

[10] G. Simon, "Discrete random variables," tech. rep., Stern School of Business, 2007. Accessed online 2012-06-27.

[11] Wikipedia, "Conditional probability." `http://en.wikipedia.org/wiki/Conditional_probability`, 2012. Accessed online 2012-11-08.

[12] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. Springer Verlag, 2007.

[13] A. Nicholson, O. Woodberry, and C. Twardy, "The "native fish" bayesian networks," tech. rep., Bayesian-Intelligence, 2010. Accessed online 2012-06-21.

[14] Wikipedia, "Inference." `http://en.wikipedia.org/wiki/Inference`, 2012. Accessed online 2012-11-20.

[15] S. Thrun, W. Burgard, and D. Fox, *Robotics*. Intelligent Robotics and Autonomous Agents Series, Mit Press, 2005.

[16] B. G. Marcot, J. D. Steventon, G. D. Sutherland, and R. K. Mccann, "Guidelines for developing and updating bayesian belief networks applied to ecological modeling and conservation," *Canadian Journal of Forest Research*, vol. 36, pp. 3063–3074, dec 2006.

[17] F. Crome, M. Thomas, and L. Moore, "A novel bayesian approach to assessing impacts of rain forest logging," *Ecological Applications*, vol. 6, no. 4, pp. 1104–1123, 1996.

[18] Y. Cohen, "Bayesian estimation of clutch size for scientific and management purposes," *Journal of Wildlife Management*, vol. 52, no. 4, pp. 787–793, 1988.

[19] T. C. Haas, "Partial validation of bayesian belief network advisory systems," *Artificial Intelligence Applications*, vol. 5, pp. 59–71, 1991.

[20] yWorks, "yed graph editor." `http://www.yworks.com/en/products_yed_about.html`, 2012. Accessed online 2012-11-08.

[21] Wikipedia, "Reference class problem." `http://en.wikipedia.org/wiki/Reference_class_problem`, 2012. Accessed online 2012-10-10.

[22] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[23] F. C. Stanley and J. Goodman, "An empirical study of smoothing techniques for language modeling," tech. rep., Harvard University and Cambridge, 1998.

[24] C. D. Manning and H. Schütze, *Foundations of Statistical Natural Language Processing*. Cambridge, MA: MIT Press, 1999.

[25] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Englewood Cliffs, NJ: Prentice Hall, 2000.

[26] S. Easterbrook, "The difference between verification and validation." `http://www.easterbrook.ca/steve/?p=2030`, 2012. Accessed online 2012-06-29.

[27] K. Korb and A. Nicholson, *Bayesian Artificial Intelligence*. Chapman and Hall, 1 ed., sep 2003.

[28] J. Rintanen, "Complexity of planning with partial observability," in *Proceedings of the 14th International Conference on Automated Planning (ICAPS'04)*, pp. 345–354, 2004. Accessed online 2012-06-21.

[29] Wikipedia, "Paraffin." `http://en.wikipedia.org/wiki/Paraffin`, 2012. Accessed online 2012-11-13.

[30] Wikipedia, "Gibibyte." `http://en.wikipedia.org/wiki/Gibibyte`, 2012. Accessed online 2012-11-13.

[31] M. Henrion, "Practical issues in constructing a bayes' belief network," in *Proceedings of the Third Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-87)*, (Corvallis, Oregon), pp. 132–139, AUAI Press, 1987.

[32] A. Onisko, M. J. Druzdzel, and H. Wasyluk, "Learning bayesian network parameters from small data sets: application of noisy-or gates," *Int. J. Approx. Reasoning*, vol. 27, no. 2, pp. 165–182, 2001.

[33] M. Krysander, *Design and Analysis of Diagnosis Systems Using Structural Methods*. PhD thesis, Linköpings universitet, June 2006.

[34] F. van Harmelen, V. Lifschitz, and B. Porter, *Handbook of Knowledge Representation*. San Diego, USA: Elsevier Science, 2007.

[35] M. Henrion and M. Morgan, *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*. Cambridge University Press, 1992.

[36] C. Huang and A. Darwiche, "Inference in belief networks: A procedural guide," *International Journal of Approximate Reasoning*, vol. 15, pp. 225–263, 1996.

[37] R. D. Shachter and M. A. Peot, "Simulation approaches to general probabilistic inference on belief networks," in *Proceedings of the Fifth Annual Conference on Uncertainty in Artificial Intelligence*, UAI '89, (Amsterdam, The Netherlands, The Netherlands), pp. 221–234, North-Holland Publishing Co., 1990.

Appendix A

Acronyms

AL	Air leakage
BN	Bayesian Network
CB	Cracked or broken
CV	Component variable
CPT	Conditional Probability Table
DTC	Diagnosis Trouble Code
ECU	Electric Control Unit
EF	Electrical fault
EMS	Engine Management System
EPIS	Evidence Pre-propagation Importance Sampling
FL	Fuel leakage
FPR	False positive rate
HP	High Pressure
HPC	High Pressure Connector
HPP	High Pressure Pump
IMV	Inlet Metering Valve
LNON	Leaky-noisy-OR-node
MPC	Maximum parent count
OV	Observation variable
NF	Non-faulty
SC	Stuck or clogged
XPI	Extreme High Pressure Fuel-Injection

Appendix B

Notation

Here follows a list of the most fundamental notation used in this thesis.

X	A stochastic variable.
$\boldsymbol{X}$	A set of stochastic variables.
$pa(X)$	List of parent nodes to X.
$pa_c(X)$	Size of parent list to X.
$\boldsymbol{M}(X)$	List of fault modes of X.
$M_c(X)$	Size of fault mode list .

Appendix C

Pseudocode

```
function generateIntermediateNodes
    bigTrees ← getBigTrees()
    for all tree ∈ bigTrees do
        leaf ← tree.getLeaf()
        nodeLayers ← getLayers(tree.getRoots())
        lowerLayer ← ∅
        upperLayer ← tree.getRoots()
        for all nodeLayer ∈ nodeLayers do
            lowerLayer ← nodeLayer
            countUpperLayer ← 0
            countLowerLayer ← 0
            child ← owerLayer.getNode(countLowerLayer)
            for all parent ∈ upperLayer do
                if child = noisyNode then  ▷ Let child be the new child of node parent, instead of leaf
Copy probabilities and fault modes
                    changeParent(parent, leaf, child)
                end if
                child.addParent(parent)
                countUpperLayer + +
                if countUpperLayer = MAX_PARENT_COUNT then
                    countUpperLayer ← 0
                    countLowerLayer + +
                    if countLowerLayer < lowerLayer.size then
                        child ← lowerLayer.getNode(countLowerLayer)
                    end if
                end if
            end for
            upperLayer ← lowerLayer
        end for
                                                    ▷ Add bottom layer as parents to leaf node
        for all parent ∈ lowerLayer do
            leaf.addParent(parent)
        end for
    end for
end function


function getBigTrees
    bigTrees ← ∅
    for all node ∈ nodesWithTooManyParents do
        tree ← setOfTrees(nodeWithTooManyParents)
        leaf ← tree.leaf
        leaf.purgeParentList()
        bigTrees ∪ tree
    end for
    return bigTrees
end function
```

```
function getLayers(rootLayer)
    layers ← ∅
    graphHeight ← getHeightOfGraph()
    rootCount ← rootLayer.size()
                                    ▷ number of extra layers, this includes layers of Noisy/OR nodes
    extraLayerCount ← ceil(logM(rootCount)) − 1          ▷ MAX_PARENT_COUNT-logarithm
    heightPerLayer ← graphHeight/(extraLayerCount + 1)
    return getLayersRecursive(postionData, layers, rootLayer)
end function


function getLayersRecursive(heightPerLayer, yCoordinateNextNode, layers, parentLayer)
    parentCount ← parentLayer
                                                          ▷ Base case for the recursion
    if parentCount ≤ MAX_PARENT_COUNT then
        return layers
    end if
    newLayer ← ∅
    nodeType ← null
    if parentLayer.nodeType = rootNode then
        nodeType ← NOISY
    else
        nodeType ← OR
    end if
    newLayerNodeCount ← ceil(parentCount/MAX_PARENT_COUNT)
    getLayerInfo ← getLayerInfo(parentLayer)
    leftmostNode ← getLayerInfo.getLeftmostNode
    xCoordinateFirstNodeInLayer ← leftmostNode.getXCoordinate()
    widthOfPreviousLayer ← getLayerInfo.getWidth()
    widthPerNode ← widthOfPreviousLayer/(newLayerNodeCount + 1)
    xCoordinateNextNode ← xCoordinateFirstNodeInLayer + widthPerNode
    for 1..newLayerNodeCount do
        newNode ← newNode(type, xCoordinateNextNode, yCoordinateNextNode)
        newLayer ∪ newNode
        xCoordinateNextNode+ = widthPerNode
    end for
    layers ∪ newLayer
    yCoordinateNextNode+ = heightPerLayer
    return getLayersRecursive(heightPerLayer, yCoordinateNextNode, layers, newLayer)
end function
```

TRITA-CSC-E 2013:008
ISRN-KTH/CSC/E--13/008-SE
ISSN-1653-5715

www.kth.se